

SDD I-Game

SOFTWARE DESIGN DOCUMENT FOR THE I-GAME
PROJECT

RICK BENIERS (STUDENT), FOUAD TAISSATE (STUDENT),
ROWIN VAN GILS (STUDENT)

SDD I-Game

SOFTWARE DESIGN DOCUMENT FOR THE I-GAME PROJECT

Rick Beniers, 1155108

Hogeschool Windesheim
AD Software Development
LABS 1

EVE-code: EVE-WFSDAD.PL1.23_2324
Docent: Stefan Hoeksema

Almere
26 januari 2025

Inhoudsopgave

Inhoudsopgave.....	2
1. Inleiding.....	1
1.1 Unified modeling language	1
1.2 Solid.....	2
1.3 Ontwerp patronen	2
1.4 Document versie beheer.....	4
2.0 Projectdefinitie	6
2.1 Probleemanalyse	6
2.2 Gekozen oplossingen & gestelde doelen.....	6
2.3 Haalbaarheid van oplossingen	6
3.0 Architectuur	7
3.0.1 Ontwerp patroon beschrijving.....	7
3.1 Epic & Userstory	8
3.2 Usecase beschrijving & usecase diagrammen	18
3.2.1 Usecase 01	18
3.2.2 Usecase 02	19
3.2.3 Usecase 03	20
3.2.4 Usecase 04	21
3.2.5 Usecase 05	22
3.2.6 Usecase 06	23
3.2.7 Usecase 7.....	24
3.2.8 Usecase 8.....	25
.....	25
3.2.9 Usecase 9.....	26
.....	26
3.2.10 Usecase 10.....	27
3.3 Toestandsdiagram.....	29
3.4 Activity diagram.....	30
3.5 Sequence diagram	31
3.6 Gegevensstromen	32
3.7 Schermontwerpen	33
3.7.1 Login scherm	33
3.7.2 Rol selectie scherm.....	35
3.7.3 Simulator scherm	37
3.7.4 Score panel.....	40

3.7.5	Bestelling geschiedenis	42
3.7.6	Administrator functies	44
4	Technische ontwerpen	46
4.1	ERD Database ontwerp	46
4.2	Class diagram	47
5	Beveiliging	51
5.1	Veiligheid risico's	51
5.2	Oplossingen	51
6	Prestatieoverwegingen	52
7	Teststrategieën	53
8	Installatie- en implementatiehandleiding	54
9	Onderhoud en toekomstige ontwikkeling	55
9.1	Afronding project	55
9.2	Toekomstige taken en/of acties	55
10	Bronnenlijst	56

1. Inleiding

Product vision: project beschrijving van het I-Game project. Dit project is het idee van een onderzoeksteam van Hogeschool Windesheim. Het onderzoeksteam doet onderzoek naar methodes om de productiviteit van logistieke planners te verhogen.

Het onderzoeksteam wil een applicatie prototype ontwikkelen die de efficiëntie van een logistieke planner meet en vastlegt. De gegenereerde statistieken zullen de logistieke planner helpen om efficiënter te werken en verbeterpunten aan te kaarten. Het applicatie prototype zal het onderzoek input geven en een sterke basis vormen voor de rest van het onderzoek.

Product goal: het applicatie prototype is een simulator spel gebaseerd op een logistiek proces. Het simulator spel geeft een speler een logistiek vraagstuk dat opgelost kan worden door een antwoord te selecteren. Het spel zal een speler door meerdere rondes testen in zijn of haar productiviteit en beoordelen op de genomen beslissingen.

Om een inschatting van de productiviteit volledig te maken zal de applicatie alle activiteit van de computermuis vastleggen. Op deze manier zal de reactietijd van de speler samen met de productiviteit en genomen beslissingen zorgen voor een compleet overzicht van de spelers' capaciteiten. De statistieken zullen weergegeven worden door een aantal diagrammen die verdere inzicht zullen geven in de verschillende datacategorieën.

1.1 Unified modeling language

Unified Modeling Language (UML) is een standaard modelleringstaal gebruikt door software developers om nieuwe softwaresystemen te visualiseren, te creëren en te documenteren ook bruikbaar bij projecten buiten programmeren. Met een UML-model verdeel je verschillende systemen tussen details en sub details om de visualisatie van het plan eenvoudiger te maken om vervolgens uit te voeren.

Er zijn twee verschillende UMLs: structurele en gedragsgeoriënteerde diagrammen. Een structureel diagram laat de connectie tussen componenten, klassen en objecten zien en hoe deze samenvallen. Een gedrag georiënteerd diagram laat zien hoe een systeem reageert op handelingen van de users.

Wij gebruiken deze modelleringstaal vanwege het efficiënte overzicht op het modelleren, plannen en stellen van prioriteiten.

1.2 Solid

Ontwerpen moeten opgesteld worden naar de vijf SOLID principes. Deze zijn als volgt: *single responsibility* (SRP), *open-closed* (OCP), *liskov substitution* (LSP), *interface segregation* (ISP) en *dependency inversion* (DIP).

De single responsibility principle constateert dat één klasse bezig is met een onderwerp, of terwijl één verantwoordelijkheid heeft, en daarom één reden heeft om te veranderen. Zo zou één potentiële verandering in de softwarespecificatie de specificatie van de klasse beïnvloeden. De open-Closed principe houdt in dat een klasse open zou moeten staan voor uitbreiding, maar gesloten voor aanpassingen. Je zou nieuwe functionaliteiten moeten kunnen toevoegen zonder de originele code aan te raken. De liskov substitutie principe gaat over objecten van een superklasse die vervangbaar moeten kunnen zijn door objecten van een subklasse, maar het programma mag niet zijn correctheid kwijtraken. De subklasse binnen het programma breidt het gedrag uit, maar beperkt het nooit. De interface segregation principle is gespecialiseerd in het onderscheiden van interfaces. Veel klant specifieke interfaces zijn beter dan één interface voor algemeen gebruik. Klanten mogen niet gedwongen worden om een functie te implementeren die ze niet nodig hebben. Tot slot dependency inversion principle die stelt dat onze klassen afhankelijk moeten zijn van interfaces van abstracte klassen, in plaats van concrete klassen en functies.

1.3 Ontwerp patronen

Voor het college Applied design patterns (ADP) worden bepaalde elementen toegevoegd aan dit ontwerpdocument. Ontwerp patronen zijn gestandaardiseerde en gestructureerde patronen die standaardoplossingen en best practises voor veel voorkomende problemen binnen object georiënteerd ontwerpen aanbiedt. Deze patronen zijn door jarenlang onderzoek van ervaren programmeurs samengesteld.

De ontwerp patronen zijn verdeeld in drie globale categorieën:

- Creational Patterns
 - Abstract factory
 - Builder
 - Factory method
 - Prototype
 - Singleton
- Concurrency patterns
 - Thread pool pattern
 - Active object pattern
 - Half-sync/Half-Async pattern
 - Monitor pattern
 - Futures and Promises pattern
 - Leader/Followers pattern
 - Actor Model pattern

- Structural Patterns
 - Adapter
 - Bridge
 - Composite
 - Decorator
 - Facade
 - Flyweight
 - Proxy
- Behavioral Patterns
 - Chain of responsibility
 - Command
 - Interpreter
 - Iterator
 - Mediator
 - Memento
 - Observer
 - State
 - Strategy
 - Template method
 - Visitor

In totaal bestaan er meer dan 23 patronen (2024) die opgedeeld zijn in de vier hierboven getoonde categorieën. Een onderzoek gepubliceerd in een boek over ontwerp patronen door vier programmeurs die de “Group of four”(GoF) wordt genoemd. In dit boek worden 23 patronen benoemd.

De creational patterns zijn ontwerp patronen die te maken hebben met object creatie mechanismes. Deze patronen bepalen flexibiliteit in het bepalen welke objecten gecreëerd moeten worden voor bepaalde scenario's.

De structural patterns zijn ontwerp patronen die het makkelijk maken om simpele en efficiënte relaties te identificeren tussen entiteiten.

De behavioural patterns zijn ontwerp patronen die te maken hebben hoe objecten met andere objecten communiceren en interacties met elkaar vormgeven.

Voor het toepassen van ontwerp patronen in dit ontwerpdocument is een ontwerpdocument gebruikt dat al grotendeels volledig was opgesteld. Normaliter is het een betere gebruik wijze om vroeg in het ontwerpproces keuzes te maken welke ontwerp patronen gebruikt gaan worden. Het is ook niet de bedoeling om de ontwerp patronen te zien als een checklist maar meer als een stuk gereedschap om een specifiek veel voorkomend probleem op te lossen.

Om ontwerp patronen toe te voegen aan dit ontwerpdocument zullen patronen worden uitgekozen die passen bij de functionaliteiten van de ontworpen elementen.

De ontwerp patronen die worden toegevoegd zullen worden beschreven bij hoofdstuk 3.0.1. Deze patronen zullen verder worden uitgewerkt in de Epics en zullen elk hun eigen usecase krijgen binnen bestaande epics.

1.4 Document versie beheer

Versie	Datum	Beschrijving
V0.1	19-02-2024	Document lay-out opgezet.
V0.2	20-02-2024	Hfd 5,6,7,8,9,10,11 en 12 add
V0.3	28-02-2024	Usecase geschreven en inleiding geschreven
V0.4	07-03-2024	Epics uitgeschreven
V0.5	09-03-2024	Userstories uitgeschreven
V0.6	12-03-2024	Usecase diagram gemaakt
V0.7	19-03-2024	Hfd 2 toegevoegd
V0.8	22-03-2024	Acceptatiecriteria toegevoegd voor US 1 tot 20
V0.9	24-03-2024	Schermontwerp hfd 3.6.1 en 3.6.2 gemaakt
V0.9.1	29-03-2024	Schermontwerp hfd 3.6.3, 3.6.4 en 3.6.5 gemaakt
V0.9.2	04-04-2024	Schermontwerp hfd 3.6.6
V0.9.3	09-04-2024	ERD v1.0 gemaakt
V0.9.4	12-04-2024	ERD v2.0 en V3.0 gemaakt
V0.9.5	13-04-2024	Use case beschrijvingen toegevoegd
V0.9.6	14-04-2024	Voor alle use case beschrijvingen use case diagrammen gemaakt en activity diagram
V1.0.1	14-04-2024	Eindoplevering Labs 1
V1.0.2	19-04-2024	Sequency diagram gemaakt
V1.0.3	23-04-2024	State diagram gemaakt
	26-04-2024	Epic en usecase 5 toegevoegd
V1.0.4	06-05-2024	Schermontwerp gemaakt voor usecase 5
V1.0.5	07-05-2024	Epic en usecase 6 gemaakt
V1.0.6	07-05-2024	Schermontwerp voor usecase 6 gemaakt.
V1.0.7	18-05-2024	Class diagram V2.0 gemaakt
V1.0.8	30-05-2024	Project definitie aangepast
V1.0.9	06-06-2024	Hfd 5 geschreven
V1.1	07-06-2024	Hfd 9, 8 en 6 geschreven
V2.0	07-06-2024	Eindoplevering Labs 2
V2.1	24-01-2025	Hoofdstuk indeling veranderd. Tekst toegevoegd aan hfd 6 & 7.
V2.2	25-01-2025	Hfd 4.2 verbeterd en tekst toegevoegd
V2.3	26-01-2025	Gegevens stromen toegevoegd aan hfd en bijschriften toegevoegd en spellingcontrole uitgevoerd.
V2.3	26-01-2025	Eindoplevering Her Labs 1

V3.0	19-01-2025	Beschrijving van ontwerp patronen toegevoegd aan de inleiding.
V3.0.1	02-02-2025	Bronnenlijst toegevoegd.
V3.0.2	03-02-2025	Uitgekozen patronen beschreven in hun eigen usecase met userstories.
V3.0.3	05-02-2025	Document structuur licht veranderd en bijna lege pagina's compacter vormgegeven.
V3.0.4	19-03-2025	Activity diagram en class diagram verbeterd en uitgebreid. Deze diagrammen waren niet compleet
V3.0.5	19-03-2025	Feedback van docent Labs 1 verwerkt. "Duidelijk aangeven waarom keuzes gemaakt zijn" voor de epics en Usecase.
V3.1.0	20-03-2025	Eindoplevering Her ADP

2.0 Projectdefinitie

2.1 Probleemanalyse

Logistieke planners en managers hun schattingen over hoeveel goederen per tijd geleverd worden zijn niet altijd accuraat. Echter, veel van deze planners zijn niet bekend met de fout van hun eigen schatting en welke methodes er bestaan om schattingen meer accuraat te maken.

2.2 Gekozen oplossingen & gestelde doelen

Voor verbetering rond het schattingsgedrag van deze planners, ontwikkelen wij een intuïtie-game (*I-game*). Doormiddel van het ophalen van data uit dit spel ontvangen de planners beter inzicht over hun schattingen en zo kunnen ze nieuwe manieren onderzoeken om deze schattingen meer accuraat te maken.

Ons doel is om deze simulatie te bouwen als webapplicatie en het front-end te designen vriendelijk genoeg voor de logistieke planners. Verder moet de applicatie kunnen communiceren met de onderliggende simulator die bereikbaar moet zijn via een API.

2.3 Haalbaarheid van oplossingen

Doormiddel van gebruik van Laravel kunnen we sneller dan gebruikelijk een applicatie ontwikkelen, dit zou daarom wel haalbaar moeten zijn. Om een inzicht te krijgen over wat user-friendly is voor een logistieke planner kunnen we navragen bij onze opdrachtgever, dus dit zou ook haalbaar moeten zijn.

3.0 Architectuur

3.0.1 Ontwerp patroon beschrijving

Singleton patroon

Het singleton patroon is een “creational” ontwerp patroon dat een gebruikswijze biedt om van een klasse altijd maar een instantie te hebben. Het singleton patroon wordt vaak gebruikt voor een klasse die maar een instantie moet hebben en een centraal punt heeft om aangeroepen te worden. Voorbeelden hiervan zijn classes die interne of externe management van de applicatie verzorgt zoals een log systeem of een window manager.

In dit project wordt het singleton patroon gebruikt om het administrator scherm van altijd een instantie te voorzien en ervoor te zorgen dat de instantie een centraal punt heeft waar zij aangeroepen kan worden. Er is voor dit patroon gekozen omdat het administrator scherm per gestarte applicatie maar 1x mag bestaan.

Er is ook overwogen om het patroon te laten toepassen op de classes van de simulator maar het centrale toegang punt voor deze classes zullen dan een probleem gaan vormen. Daarom is besloten om dit patroon alleen toe te passen op de administrator klasse.

Het singleton patroon werkt door een klasse een referentie naar zichzelf te geven en de constructor “Private” te maken zodat deze alleen in de eigen klasse aan te roepen is. Er moet een operatie worden toegevoegd die een check uitvoert of de instantie van de klasse al bestaat of niet. Als de instantie wordt aangeroepen wordt deze door dezelfde operatie teruggestuurd.

Dit patroon wordt behandeld in usecase 8.

Chain of responsibility Method patroon

Dit patroon biedt een gebruikswijze voor het afhandelen van aanvragen binnen een applicatie doormiddel van een “Handler” interface en een of meerdere subklassen hiervan die de operatie(s) uit de interface kunnen overschrijven en invullen zoals ze zelf willen. Op deze manier is er een centraal punt voor het afhandelen van aanvragen.

In dit project wordt dit patroon toegepast voor het simulator spel waarin spelers aan elkaar aanvragen kunnen sturen waarmee ze producten kunnen bestellen of versturen.

Dit patroon is gekozen om verzoeken tussen spelers(instanties) efficiënt af te handelen in plaats van overal en nergens. Dit patroon wordt behandeld in usecase 9.

Factory Method patroon

Het Factory method patroon biedt een gebruikswijze om instantie van classes te maken. Een interface wordt gebruikt om objecten te instantiëren doormiddel van subklassen. De gecreëerde objecten kunnen gerefereerd worden door een centrale interface.

In dit project wordt dit patroon gebruikt om op een gestructureerde en geordende manier objecten te creëren zoals de verschillende elementen van de applicatie. Dit patroon wordt behandeld in usecase 10.

3.1 Epic & Userstory

Epic 01: Gebruiker data verzamelen (could have)

Usecase 1:

Aan het einde van het spel krijg ik als speler een kort overzicht van mijn prestaties te zien. Dit overzicht bevat statistieken zoals mijn reactietijd en aantal interacties, ook bevatten de resultaten statistieken zoals gemiddelde reactietijden en gemiddeld aantal interacties van mij en de andere spelers. Daarnaast is er een scoreboard dat de scores en rangschikking van alle deelnemers weergeeft gebaseerd op het verdiende bedrag dat is verdiend door het produceren/verwerken en leveren van producten.

Er is gekozen om deze usecase te ontwerpen, in overleg met de stakeholders en opdrachtgever, zodat de applicatie data van de spelers verzameld zodat een indicatie gemaakt kan worden van de capaciteiten van de spelers. De verzamelde data worden aan het einde van de simulator weergegeven aan de speler. De opdrachtgever heeft opdracht gegeven om dit scherm te ontwerpen voor de speler. De opdrachtgever heeft aangegeven dat de opdrachtgever de data ook direct uit de database kan ophalen zodat de opdrachtgever de optie heeft om de data zelf te analyseren.

- **US19:** Als speler wil ik op het einde van een spel mijn reactietijd kunnen inzien, zodat ik kan zien hoe snel ik mijn beslissingen heb genomen.
- **US20:** Als speler wil ik op het einde van een spel mijn interacties kunnen inzien, zodat ik weet wat voor beslissingen ik heb genomen tijdens het spel.
- **US21:** Als speler wil ik een scoreboard zien aan het einde van het spel, zodat ik kan zien wie het beste heeft gespeeld.
- **US22:** Als speler wil ik een gemiddelde reactietijd zien aan het einde van het spel, zodat ik kan zien hoe snel de gemiddelde beslissing werd genomen.
- **US23:** Als speler wil ik een gemiddelde aantal interactie zien aan het einde van het spel, zodat ik kan zien hoeveel acties er werden genomen in het spel.
- **US43:** Als speler wil ik een bedrag zien dat ik heb verdiend doormiddel van het produceren, verwerken en leveren van producten, zodat ik kan zien hoeveel profijt/verlies ik heb gekregen.
- **US51:** Als speler wil ik zien hoeveel de gehele supplychain aan bedrag heeft verdiend, zodat ik een inschatting van mijn prestaties kan maken.
- **US52:** Als speler wil ik door de lijst met spelers kunnen scrollen, zodat ik kan zien wie er allemaal meedoet.

Epic 02: Simulator spel design A (Could have)

Usecase 2:

Als speler wil ik producten kunnen bestellen via het invoeren van een cijfer vervolgd door het beantwoorden van een vraagstuk. Ook wil ik een weergave van de levertijd, aantal gespeelde rondes en de eenheden die het product reflecteren. Tot slot wil ik een weergave van het beheer van logistieke processen zien die verdeeld zijn over verschillende bedrijfstakken.

De opdrachtgever heeft gevraagd om twee verschillende usecases te ontwerpen die de UI van de simulator weergeven. Het ontwerp moet de style van het populaire logistieke spel *"The bear game"* overnemen. De spelers moeten gebaseerd op de data die de UI weergeeft beslissingen kunnen maken over het bestellen en produceren van producten. Dit ontwerp verschilt met de het tweede ontwerp doormiddel van de data die wordt weergegeven.

In ontwerp A wordt een scherm aan de speler getoond die voor alle spelers hetzelfde is. Alle spelers kunnen de weergegeven data van de andere spelers duidelijk zien zodat de UI daadwerkelijk een logistiek proces weergeeft met verschillende producten en bedrijven met verschillende rollen.

- **US06:** Als speler wil ik een cijfer kunnen invoeren, zodat ik producten kan bestellen.
- **US07:** Als speler wil ik via een knop de vraag kunnen beantwoorden, zodat ik een bestelling kan plaatsen.
- **US08:** Als speler wil ik de levertijd van een product kunnen zien, zodat ik een betere inschatting kan maken over hoelang het leveren duurt.
- **US09:** Als speler wil voor elk product eenheden zien die het product reflecteren, zodat ik een betere inschatting kan maken over mijn prestaties.
- **US10:** Als speler wil ik zien hoeveel rondes er gespeeld zijn en hoeveel rondes er nog komen in het huidige spel, zodat ik een duidelijk beeld krijg over hoelang het proces duurt.
- **US11:** Als speler wil ik vier verschillende bedrijfstakken zien in het logistieke proces, zodat ik weet welke partijen van toepassing zijn.
- **US12:** Als speler ruiker wil ik voor elk bedrijfstak zien hoeveel producten er besteld zijn, zodat ik weet hoeveel producten ik moet leveren.
- **US13:** Als speler wil ik voor elk bedrijfstak zien hoeveel producten er geleverd moeten worden, zodat ik kan filteren hoeveel elke tak moet ontvangen.
- **US14:** Als speler wil ik een cijfers kunnen invoeren, zodat ik de producten kan leveren.

Epic 03: Gebruiker registreren (must have)

Usecase 3:

Als gebruiker moet ik een naam invullen om het simulator spel te beginnen. Als eerste vul ik mijn gekozen naam in het tekst vak in waarboven het woord gebruikersnaam staat. Als tweede vul ik de gamecode in het juiste tekst vak in. De gamecode heb ik zojuist ontvangen van de administrator van het spel.

Als derde klik ik op de knop registreren om mijn gekozen naam op te slaan. Wanneer de naam succesvol is geregistreerd word ik doorgestuurd naar het rol selectie scherm van het simulator spel.

Als de naam al in gebruik is zal na het drukken op de registreren knop een rood veld getoond worden dat de tekst "Gebruikersnaam is al geregistreerd" bevat. Als de gebruikersnaam langer dan 10 karakters is zal een rood veld getoond worden dat de tekst "Gebruikersnaam te lang. Maximaal 10 karakters toegestaan" bevat.

In overleg met de opdrachtgever is er een ontwerp gemaakt van een scherm waarop de speler zich kan registreren. In overleg met de opdrachtgever is er gekozen om het registratie proces niet met een wachtwoord vorm te geven maar om alleen een gebruikersnaam en gamecode vorm te geven voor het registratie proces. Dit maakte het registreren makkelijk en snel. De speler ontvangt van de administrator de gamecode en kan zelf een gebruikersnaam bedenken en invullen.

- **US01:** Als speler wil ik mijn gebruikersnaam kunnen invullen in een tekst vak, zodat de prestaties die ik behaal aan mij gekoppeld kunnen worden.
- **US02:** Als speler wil ik op de registreren knop kunnen drukken, zodat mijn ingevulde gebruikersnaam wordt opgeslagen.
- **US03:** Als speler wil ik een rood veld getoond krijgen, zodat ik kan zien dat mijn gekozen gebruikersnaam al in gebruik is.
- **US04:** Als speler wil ik een rood veld getoond krijgen, zodat ik kan zien dat mijn gebruikersnaam te lang is.
- **US05:** Als speler word ik na het succesvol registreren doorgestuurd worden naar het rolselectie scherm, zodat ik een rol kan selecteren.
- **US72:** Als speler wil ik een gamecode kunnen invullen zodat ik gekoppeld kan worden aan de simulator instantie.

Epic 04: Simulator spel design B (must have)

Usecase 4:

Als speler wil ik een overzicht zien van alle gegevens die belangrijk zijn voor de bedrijfstuk die ik uitgekozen heb.

Ik wil als speler zien hoeveel producten er besteld worden bij mijn bedrijf, hoeveel producten op voorraad zijn, hoeveel producten op dit moment verstuurd worden, hoeveel producten niet verstuurd kunnen worden door te weinig voorraad en hoeveel producten ontvangen worden door mijn bedrijf.

Ook wil ik als speller een aantal producten kunnen bestellen. De levertijd van producten moet worden weergegeven evenals de huidige spel ronde. Wanneer het spel start is mijn voorraad nul. Een aantal producten zal worden besteld bij mijn bedrijf, de levertijd van deze producten geeft aan wanneer ik deze producten moet leveren. Gebaseerd op deze informatie moet ik een aantal producten bestellen zodat ik deze kan verwerken in mijn bedrijf en uiteindelijk kan versturen naar mijn klanten.

Als er geen bestellingen zijn gedaan door klanten maar ik wel producten bestel zal mijn voorraad omhooggaan. Als ik geen producten bestel maar er wel bestellingen van klanten binnen komen zal mijn voorraad afnemen. Als ik geen voorraad meer heb zal de bestelling van de klant op pauze worden gezet en worden verstuurd zo snel als er producten bij mij binnen komen. De bestelling van de klant is op dat moment een "Back order". Voor elk product dat ik lever aan mijn klant krijg ik een geldbedrag. Dit bedrag wordt toegevoegd aan mijn bankrekening. Als ik producten bestel van een ander bedrijf moet daar een bedrag voor betaald worden. Dit bedrag wordt afgetrokken van mijn bankrekening. Mijn bankrekening wordt weergegeven door een geldbedrag op het scherm zodat dir altijd inzichtelijk is.

De opdrachtgever heeft gevraagd om twee verschillende usecases te ontwerpen die de UI van de simulator weergeven. Het ontwerp moet de style van het populaire logistieke spel "*The bear game*" overnemen. Er zijn twee verschillende ontwerpen gemaakt voor de UI van de simulator. In ontwerp B wordt data weergegeven aan de speler die ook alleen voor die specifieke speler bedoeld is. Dit verschilt met ontwerp A. In ontwerp A wordt het complete logistieke proces weergegeven en is er meer data beschikbaar voor de speler. In opdracht van de opdrachtgever is er een tweede ontwerp gemaakt.

In ontwerp B wordt niet het complete logistiek proces weergegeven maar alleen de data voor de specifieke spelers rol wordt aan de speler getoond. In overleg met de opdrachtgever is uiteindelijk gekozen om ontwerp B te realiseren. Hier is voor gekozen omdat de opdrachtgever het beter vond dat alle spelers alleen data die belangrijk is voor zichzelf getoond kregen. Dit is tegelijkertijd ook een realistischer beeld omdat in de logistiek het niet altijd mogelijk is om een compleet beeld van het logistieke proces te hebben.

Op de volgende pagina worden de specifieke userstories weergegeven.

- **US29:** Als speler wil ik aan het begin zien welke rol ik heb in het spel, zodat ik betere ik weet welke rol ik heb en welke taak ik verantwoordelijk voor ben.
- **US30:** Als speler wil ik mijn gebruikersnaam kunnen zien tijdens het spelen van het spel, zodat ik weet met welke naam ik het spel speel.
- **US31:** Als speler wil ik zien hoeveel klanten een bestelling hebben aangemaakt, zodat ik weet hoeveel vracht ik nodig ga hebben.
- **US32:** Als speler wil ik zien hoeveel voorraad mijn bedrijf heeft, zodat ik weet hoeveel producten ik moet bestellen.
- **US33:** Als speler wil ik producten kunnen bestellen, zodat ik deze kan verwerken en versturen naar mijn klanten.
- **US34:** Als speler wil ik zien hoeveel levertijd er is voor de producten die ik bestel, zodat ik weet hoelang het duurt voordat mijn bestelling aankomt.
- **US35:** Als speler wil ik zien hoeveel bestellingen op pauze staan, zodat ik weet welke bestellingen ik vroeg of laat moet voortzetten.
- **US36:** Als speler wil ik zien hoeveel spel rondes er momenteel zijn afgerond, zodat ik weet hoever ik ben met het spel.
- **US37:** Als speler wil ik zien hoeveel producten er aan mij geleverd worden, zodat ik weet hoeveel producten ik ontvang.
- **US44:** Als speler wil ik zien hoeveel geld ik bezit, zodat ik weet hoeveel producten ik kan bestellen.
- **US45:** Als speler wil ik een bedrag ontvangen als ik een product geleverd heb, zodat ik een beloning krijg voor mijn uitgevoerde werk.
- **US46:** Als speler wil ik een bedrag betalen als ik een product bestel bij een ander bedrijf, zodat deze tak een reden heeft om het product aan mij te leveren.

Epic 05: simulator spel rol selectie (must have)

Usecase 5:

Als speler van het simulator spel wil ik na het registreren mijn rol in het spel kunnen kiezen. Ik kan kiezen uit: *Factory*, *Distributor*, *Wholesaler* of *Retailer*. Na het selecteren en opslaan van mijn rol word ik doorgestuurd naar het simulator spel.

In overleg met de opdrachtgever is gekozen om een scherm te ontwerpen waardoor de speler een rol kan kiezen. De beschikbare rollen worden op dit scherm weergegeven. In opdracht van de opdrachtgever is er gekozen om dit een apart scherm te maken in plaats van een selecteerbare lijst met knoppen die bijvoorbeeld op het registratie scherm.

- **US16:** Als speler wil ik minimaal uit vier bedrijf takken kunnen kiezen, zodat ik verschillende rollen kan spelen.
- **US17:** Als speler wil ik mijn keuze kunnen bevestigen, zodat ik niet de verkeerde rol selecteer.
- **US18:** Als speler wil ik naar het simulator spel kunnen navigeren, zodat ik elk onderdeel binnen de simulatie makkelijk kan bereiken.
- **US38:** Als speler wil ik de rol van Factory kunnen kiezen, zodat ik de producten kan ontwikkelen.
- **US39:** Als speler wil ik de rol van Distributor kunnen kiezen, zodat ik de producten kan verzenden.
- **US40:** Als speler wil ik de rol van Wholesaler kunnen kiezen, zodat ik de producten grootschalig kan verkopen.
- **US41:** Als speler wil ik de rol van Retailer kunnen kiezen, zodat ik producten kan kleinschalig kan verkopen.

Epic 06: Administrator functies (could have)

Usecase 6:

Als administrator wil ik een spel kunnen aanmaken. Ik kan het aantal rondes invoeren en de beschikbare rollen kunnen selecteren. Er wordt een spel code gegenereerd die de spellers kunnen invoeren bij het registreren zodat de spellers aan het juiste spel deelnemen.

In opdracht van de opdrachtgever is er een ontwerp gemaakt voor een administrator scherm waardoor een administrator een aantal opties heeft om de simulatie te bewerken. In overleg met de opdrachtgever is er gekozen om de administrator de opties te geven om een maximaal aantal spel rondes in te voeren, een gamecode te genereren, om specifieke rollen beschikbaar te maken of voor de spelers te verbergen en als laatste een lijst met spelers te kunnen zien op het administrator scherm.

Een optie was om de lijst met spelers de opties te geven om spelers te verwijderen maar in overleg met de opdrachtgever is besloten om dit te ontwerpen als er tijd beschikbaar was na het implementeren van de simulator en haar functionaliteiten.

- **US24:** Als administrator wil ik een spel kunnen aanmaken, zodat de spelers toegang krijgen tot de simulatie.
- **US25:** Als administrator wil ik een spel kunnen starten, zodat de spelers het spel kunnen beginnen.
- **US26:** Als administrator wil ik een spel code kunnen zien, zodat de spelers weten welk spel ze moeten betreden.
- **US27:** Als administrator wil ik het aantal spel rondes kunnen instellen, zodat ik kan bepalen hoelang het spel duurt.
- **US28:** Als administrator wil ik instellen welke bedrijfstakken de spelers kunnen uitkiezen, zodat ik kan kiezen welke partijen op het moment beschikbaar zijn.
- **US42:** Als administrator wil ik een lijst met de spelers kunnen zien voordat het spel begint, zodat ik weet welke en hoeveel spelers er zijn.

Epic 07: Bestelling geschiedenis (will have)

Usecase 7:

Als speler wil ik tijdens het spelen van het spel een scherm kunnen openen waar ik mijn afgeronde bestellingen kan zien en de bestellingen over de hele logistieke supplychain.

Er wordt een bestelling aangemaakt voor mij als speler als ik een bestelling binnen krijg. De bestelling is afgerond als ik de bestelde producten heb geleverd. Ik kan wisselen tussen de bestellingen van mij als speler en de bestellingen van de gehele supplychain. Er wordt een bestelling voor de gehele supplychain aangemaakt als het laatste bedrijf in de supplychain een bestelling ontvangt. De bestelling is voltooit als de bestelde producten geleverd of verkocht zijn.

In overleg met de opdrachtgever is er een scherm ontworpen om de speler van data te voorzien die een overzicht geven van de eerder geplaatste bestellingen en voltooide bestellingen zowel voor de speler als voor het complete logistieke proces. De data die dit scherm weergeeft moeten voor de speler hetzelfde effect hebben als het inzien van de financiën van bijvoorbeeld een bedrijf. Het idee achter dit ontwerp is dat de speler bepaalde informatie kan verkrijgen uit de eerder geplaatste bestellingen maar de speler moet wel opletten dat de speler niet een eenzijdig beeld voorzichtzelf creëert.

De data op dit scherm biedt een aantal puzzelstukjes voor de speler om de juiste beslissingen te maken maar de speler moet wel de andere data op het simulator scherm overwegen is zijn beslissingen.

- **US47:** Als speler wil ik een scherm kunnen openen tijdens de simulator, zodat ik mijn bestellingen kan zien.
- **US48:** Als speler wil ik het verdiende bedrag, de levertijd, aantal en ronde per afgeronde bestelling zien, zodat ik weet hoeveel ik verdiend heb.
- **US49:** Als speler wil ik een scherm kunnen openen tijdens de simulator, zodat ik de bestellingen van de gehele supplychain kan zien.
- **US50:** Als speler wil ik het totaal verdiende bedrag van een bestelling van de gehele supplychain zien, zodat ik kan inschatten hoe efficiënt de gehele supplychain is.

Epic 8: Specifieke ontwerp patronen (could have)

Deze Epic en de resulterende usecases zijn ontworpen om ontwerp patronen te implementeren in dit project. De ontwerp patronen bieden een specifieke bewezen gebruikswijze zodat nieuwe software niet opnieuw mechanismen hoeft uit te vinden.

Er is gekozen voor ontwerp patronen die veelgebruikt worden in softwareontwikkeling om de gebruikswijze van de patronen aan te tonen. Er is ook overwogen om extra ontwerp patronen te ontwerpen maar uiteindelijk is voor de hieronder beschreven patronen gekozen om het principe van ontwerp patronen toe te passen op een efficiënte en in mindere mate complexe applicatie.

Usecase 8:

Als administrator wil ik het administratie scherm bij het laden van de applicatie instantiëren en voorkomen dat dit meerdere keren gebeurt. Het administrator scherm moet altijd maximaal een object hebben en dit object moet een operatie hebben waarmee het object globaal aangeroepen kan worden.

- **US53:** Het administrator scherm moet in maximaal een klasse vormgegevens worden.
- **US54:** Het administrator scherm moet een operatie bevatten die de unieke instantie van de klasse (Object) returned.
- **US55:** Het administrator scherm mag niet aanroep baar zijn doormiddel van de constructor.
- **US56:** Het administrator scherm moet een operatie bevatten die een check uitvoert of een instantie (Object) van de klasse al bestaat.
- **US57:** Het administrator scherm moet een operatie bevatten die een nieuwe instantie creëert als deze nog niet bestaat en wordt aangeroepen.

Usecase 9:

Als speler wil ik tijdens het spelen/simuleren van de applicatie aanvragen versturen naar andere spelers die een andere rol in het logistiek proces van de simulatie hebben door het chain of responsibility patroon te gebruiken.

Een aanvraag kan bestaan uit een “Buy order” of een “Sell order”. De andere speler krijgt deze aanvraag te zien en kan gebaseerd op de aanvraag beslissingen maken.

Een aanvraag wordt verstuurd door de instantie (Object) van een speler naar de interface die bepaald welk specifiek object de aanvraag gaat afhandelen.

- **US58:** Een interface voor het afhandelen van aanvragen moet gemaakt worden.
- **US59:** De interface moet door elke instantie van een speler aanroep baar zijn.
- **US60:** Elke instantie van een speler moet de operatie van de interface overnemen en voor zichzelf invullen(inheritance).
- **US61:** Elke instantie van een speler moet een “Sell order” kunnen ontvangen.
- **US62:** Elke instantie van een speler moet een “Buy order” kunnen ontvangen.
- **US63:** Elke instantie die een aanvraag kan afhandelen controleert voor elke aanvraag of dit het geval is. Zo niet dan wordt de aanvraag doorgestuurd naar de volgende instantie die een aanvraag kan afhandelen.
- **US64:** De interface moet een aanvraag die niet afgehandeld kan worden opvangen.

Usecase 10:

Als gebruiker moet een instantie van het register scherm worden gemaakt doormiddel van een “Factory” die alle instantie/object creatie behandelt en afhandelt in de applicatie.

Een abstracte klasse bevat operaties die een creator klasse kan gebruiken en overschrijven. De abstracte klasse moet alle operaties bevatten die de creator klasse moet kunnen gebruiken. De creator klasse maakt gebruik van een interface voor elke mogelijk te instantiëren klasse. De creator klasse gebruikt de interface als blauwdruk voor een te instantiëren klasse.

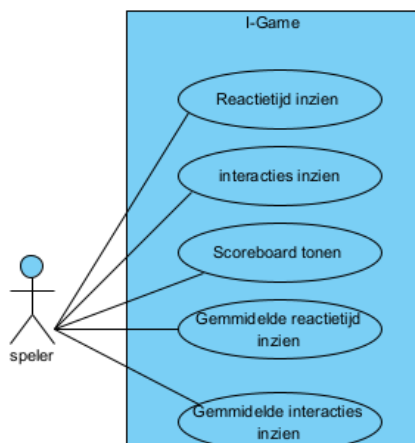
Wanneer een object gemaakt moet worden moet de creator klasse aangeroepen worden en de juiste methode aangeroepen worden.

- **US65:** Er moet een interface gemaakt worden voor de registrationView klasse.
- **US66:** Er moet een interface gemaakt worden voor de roleSelectionView klasse.
- **US67:** Er moet een interface gemaakt worden voor de simulatorView klasse.
- **US68:** Er moet een interface gemaakt worden voor de administratorView klasse.
- **US69:** Er moet een interface gemaakt worden voor de scoreboardView klasse.
- **US70:** Er moet een interface gemaakt worden voor de orderHistoryView klasse.
- **US71:** Er moet een interface gemaakt worden voor de HeaderView klasse.
- **US72:** Een abstracte klasse moet vormgegevens worden met operaties voor het creëren van een objecten gebaseerd op de interfaces.
- **US73:** Een creator klasse moet worden vormgegeven die operaties van de abstracte klasse kan implementeren en overschrijven als methoden.

3.2 Usecase beschrijving & usecase diagrammen

3.2.1 Usecase 01

Usecase 1	Gebruiker data verzamelen
Omschrijving	Na het spelen van het simulator spel wordt de verzamelde data getoond aan de speler doormiddel van een scoreboard.
Pre-conditie	Het simulator spel is afgerond en de speler heeft deelgenomen
Post-conditie	De speler wordt doorgestuurd naar de registratie pagina
Actoren	De speler
Main course of action	Aan het einde van het spel krijg ik als speler een kort overzicht van mijn prestaties te zien. Dit overzicht bevat statistieken zoals mijn reactietijd en aantal interacties, ook bevatten de resultaten statistieken zoals gemiddelde reactietijden en gemiddeld aantal interacties van mij en de andere spelers. Daarnaast is er een scoreboard dat de scores en rangschikking van alle deelnemers weergeeft.
Excepties	Getallen onder de nul zullen worden weergegeven als 0. Getallen boven de 10.000 zullen worden weergegeven als 10k of bijvoorbeeld 10.1k voor 10.100

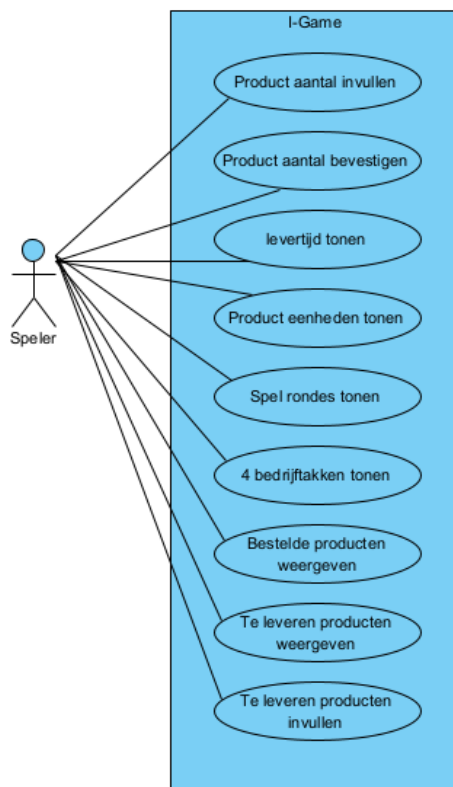


Figuur 1. Usecase diagram van usecase 1

Het bovenstaande usecase diagram bevat de user story's van usecase 1.

3.2.2 Usecase 02

Usecase 2	Simulator spel (A)
Omschrijving	De speler neemt deel aan het simulator spel na het registreren en selecteren van een rol. Tijdens het simulator spel zal de speler het juiste aantal producten moeten bestellen zodat deze verwerkt kunnen worden en uiteindelijk verzonden worden naar de klanten.
Pre-condition	De speler heeft zich geregistreerd en een rol geselecteerd
Post-condition	Na het uitspelen van het spel wordt de speler doorgestuurd naar een scoreboard
Actoren	De speler
Main course of action	Als speler wil ik producten kunnen bestellen via het invoeren van een cijfer vervolgd door het beantwoorden van een vraagstuk. Ook wil ik een weergave van de levertijd, aantal gespeelde rondes en de eenheden die het product reflecteren. Tot slot wil ik een weergave van het beheer van logistieke processen verdeeld over verschillende bedrijfstakken.
Exceptions	Negatieve getallen zullen worden omgezet naar 0. Getallen onder de nul kunnen niet ingevoerd worden.

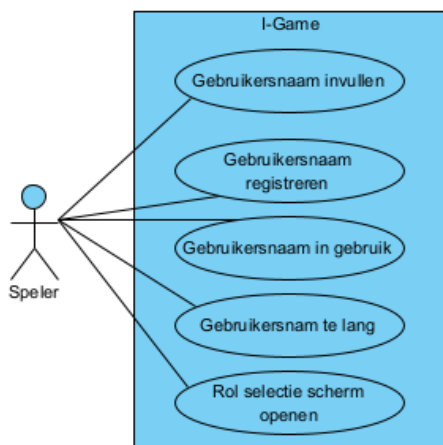


Het usecase diagram hiernaast bevat de user story's van usecase 2.

Figuur 2. Usecase diagram van usecase 2

3.2.3 Usecase 03

Usecase 3	Gebruiker registreren
Omschrijving	De speler moet de mogelijkheid hebben om een account aan te kunnen maken.
Pre-condition	Een unieke gebruikersnaam.
Post-condition	Na succesvolle registratie doorverwijzing naar rolselectie scherm.
Actoren	De speler
Main course of action	<p>Als speler moet ik een naam invullen om het simulator spel te beginnen. Als eerste vul ik mijn gekozen naam in het tekst vak in waarboven het woord gebruikersnaam staat. Als tweede druk ik op de knop registreren om mijn gekozen naam op te slaan. Wanneer de naam succesvol is geregistreerd word ik doorgestuurd naar het rol selectie scherm van het simulator spel.</p> <p>Als de naam al in gebruik is, zal na het drukken op de registreren knop een rood veld getoond worden dat deze tekst: "Gebruikersnaam is al geregistreerd" weergeeft. Als de gebruikersnaam langer dan 10 karakters is zal een rood veld getoond worden dat de tekst: "Gebruikersnaam te lang, maximaal 10 karakters toegestaan" bevat.</p>
Exceptions	<p>Als je een al gebruikte naam invult, zal je een andere naam moeten kiezen.</p> <p>Als je een te lange naam invult, zal je een kortere naam moeten kiezen.</p>

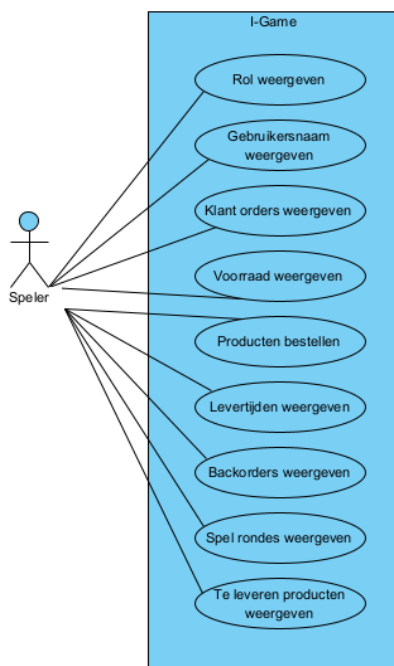


Het usecase diagram hiernaast bevat de user story's van usecase 3.

Figuur 3. Usecase diagram van usecase 3.

3.2.4 Usecase 04

Usecase 4	Simulator spel (B)
Omschrijving	De speler neemt deel aan het simulator spel na het registreren en selecteren van een rol. Tijdens het simulator spel zal de speler het juiste aantal producten moeten bestellen zodat deze verwerkt kunnen worden en uiteindelijk verzonden worden naar de klanten.
Pre-condition	De speler heeft zich geregistreerd en een rol geselecteerd
Post-condition	Na het uitspellen van het spel wordt de speler doorgestuurd naar een scoreboard
Actoren	De speler
Main course of action	<p>Als speler wil ik een overzicht zien van alle gegevens die belangrijk zijn voor de bedrijfstak die ik uitgekozen heb.</p> <p>Ik wil als speler zien hoeveel producten er besteld worden bij mijn bedrijf, hoeveel producten op voorraad zijn, hoeveel producten op dit moment verstuurd worden, hoeveel producten niet verstuurd kunnen worden door te weinig voorraad en hoeveel producten ontvangen worden door mijn bedrijf.</p> <p>Ook wil ik als speler een aantal producten kunnen bestellen. De levertijd van producten moet worden weergegeven evenals de huidige spel ronde.</p> <p>Wanneer het spel start is mijn voorraad nul. Een aantal producten zal worden besteld bij mijn bedrijf, de levertijd van deze producten geeft aan wanneer ik deze producten moet leveren. Gebaseerd op deze informatie moet ik een aantal producten bestellen zodat ik deze kan verwerken in mijn bedrijf en uiteindelijk kan versturen naar mijn klanten.</p> <p>Als er geen bestellingen zijn gedaan door klanten maar ik wel producten bestel zal mijn voorraad omhooggaan. Als ik geen producten bestel maar er wel bestellingen van klanten binnen komen zal mijn voorraad afnemen. Als ik geen voorraad meer heb zal de bestelling van de klant op pauze worden gezet en worden verstuurd zo snel als er producten bij mij binnen komen.</p>
Exceptions	Het niet invullen van een order betekend dat de order wordt omgezet naar 0. Getallen onder de nul kunnen niet ingevoerd worden.

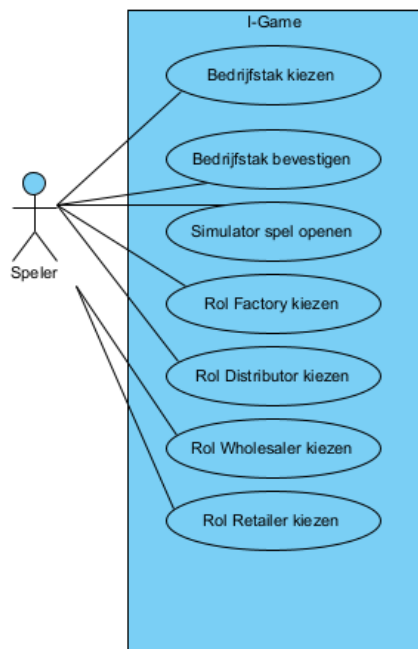


Het usecase diagram hiernaast bevat de user story's van usecase 4.

Figuur 4. Usecase diagram van usecase 4

3.2.5 Usecase 05

Usecase 5	Simulator spel rol selectie
Omschrijving	Na het registreren wil ik kunnen kiezen tussen de rollen: Factory, Distributor, Wholesaler of Retailer.
Pre-condition	Geregistreerd zijn met een gebruikersnaam.
Post-condition	Na het kiezen van een rol doorverwezen naar het simulator spel.
Actoren	De speler
Main course of action	Als speler van het simulator spel wil ik na het registreren mijn rol in het spel kunnen kiezen. Ik kan kiezen uit: Factory, Distributor, Wholesaler of Retailer. Na het selecteren en opslaan van mijn rol word ik doorgestuurd naar het simulator spel.
Exceptions	Een mogelijkheid voor de administrator om een rol aan of uit te zetten.

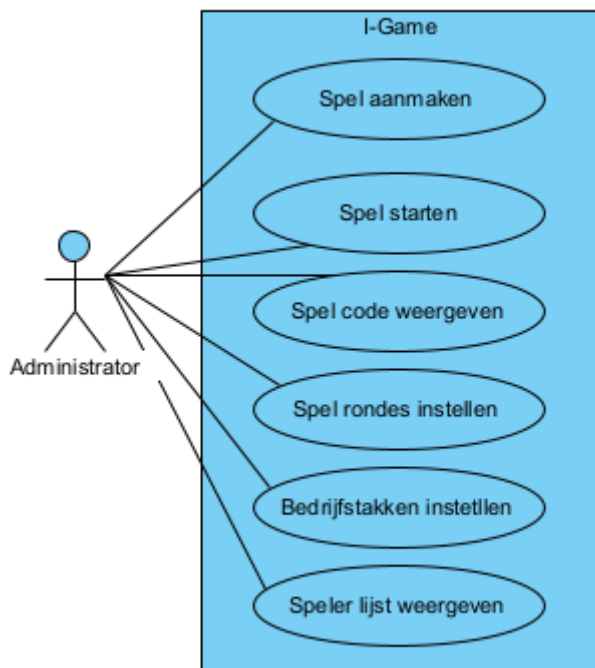


Het usecase diagram hiernaast bevat de user story's van usecase 5.

Figuur 3-5. Usecase diagram van usecase 5.

3.2.6 Usecase 06

Usecase 6	Administrator functies.
Omschrijving	Als administrator moet je de volledige macht kunnen hebben over de applicatie, rollen en de spelers.
Pre-condition	De mogelijkheid om in te loggen als administrator.
Post-condition	Mogelijkheid om de applicatie te besturen naar jouw wens.
Actoren	De administrator.
Main course of action	Als administrator wil ik: <ul style="list-style-type: none">• Een spel code genereren die de spelers kunnen invoeren bij het registreren zodat de spelers aan het juiste spel deelnemen.• Het aantal rondes invoeren• De beschikbare rollen kunnen selecteren.
Exceptions	Maximaal 1 administrator per spel.

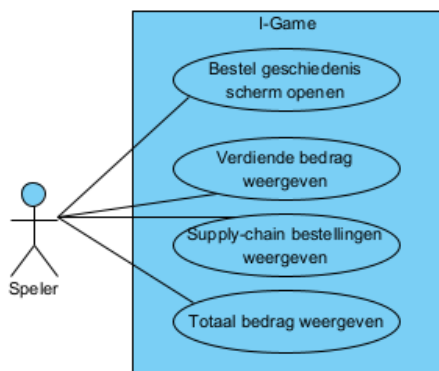


Figuur 6 bevat de user story's van usecase 6.

Figuur 3-6. Usecase diagram van usecase 6.

3.2.7 Usecase 7

Usecase 7	Bestelling geschiedenis.
Omschrijving	Als speler wil ik de geschiedenis van de bestellingen kunnen weergeven.
Pre-condition	Aanmaak van bestelling wanneer je een bestelling binnen krijgt.
Post-condition	Levering van bestelde producten.
Actoren	De speler
Main course of action	<p>Als speler wil ik tijdens het spelen van het spel een scherm kunnen openen waar ik mijn afgeronde bestellingen kan zien en de bestellingen over de hele logistieke supplychain.</p> <p>Er wordt een bestelling aangemaakt voor mij als speler als ik een bestelling binnen krijg. De bestelling is afgerond als ik de bestelde producten heb geleverd. Ik kan wisselen tussen de bestellingen van mij als speler en de bestellingen van de gehele supplychain. Er wordt een bestelling voor de gehele supplychain aangemaakt als het laatste bedrijf in de supplychain een bestelling ontvangt. De bestelling is voltooit als de bestelde producten geleverd of verkocht zijn.</p>
Exceptions	Alleen de geschiedenis van het huidige spel moet weergegeven worden.

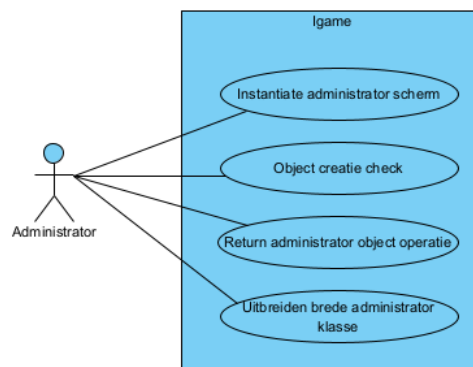


Figuur 7: Usecase diagram van usecase 7

Het usecase diagram hiernaast bevat de user story's van usecase 7.

3.2.8 Usecase 8

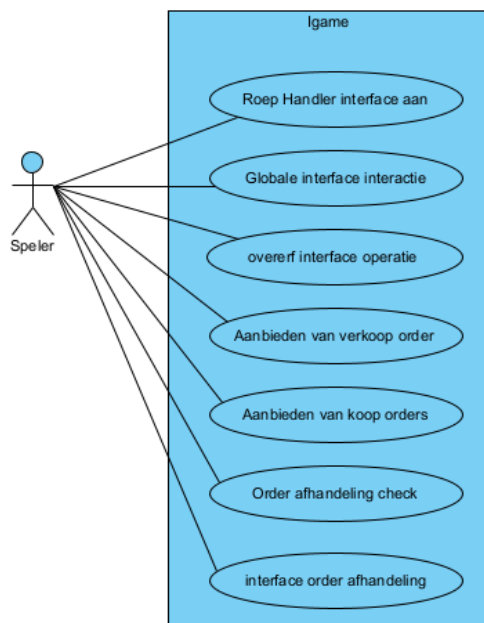
Usecase 8	Singleton ontwerp patroon
Omschrijving	Als administrator wil ik maximaal 1 instantie per applicatie hebben van het administrator scherm.
Pre-condition	De applicatie is gestart
Post-condition	Geen
Actoren	Administrator
Main course of action	<p>Het Singleton patroon functioneert door de klasse die het singleton patroon moet implementeren een extra methode te geven. Deze methode is static en kan net als een constructor overall worden aangeroepen. De methode voert een check uit of er al een object van de administratorView klasse bestaat. Als er al een object bestaat dan returned de methode een referentie naar het object. Wanneer het object nog niet bestaat dan maakt de methode een object van de klasse door de administratorView te instantiëren en returned een referentie naar het zojuist gemaakte object.</p> <ul style="list-style-type: none"> • US53: Het administrator scherm moet in maximaal een klasse vormgegevens worden. • US54: Het administrator scherm moet een operatie bevatten die de unieke instantie van de klasse (Object) returned. • US55: Het administrator scherm mag niet aanroep baar zijn doormiddel van de constructor. • US56: Het administrator scherm moet een operatie bevatten die een check uitvoert of een instantie (Object) van de klasse al bestaat. • US57: Het administrator scherm moet een operatie bevatten die een nieuwe instantie creëert als deze nog niet bestaat en wordt aangeroepen.
Exceptions	Een nieuwe aanvraag voor het maken van een instantie van de administratorView resulteert in een referentie naar het huidige geïnstantieerde administratorView object.



Figuur 21. Usecase diagram, Singleton.

3.2.9 Usecase 9

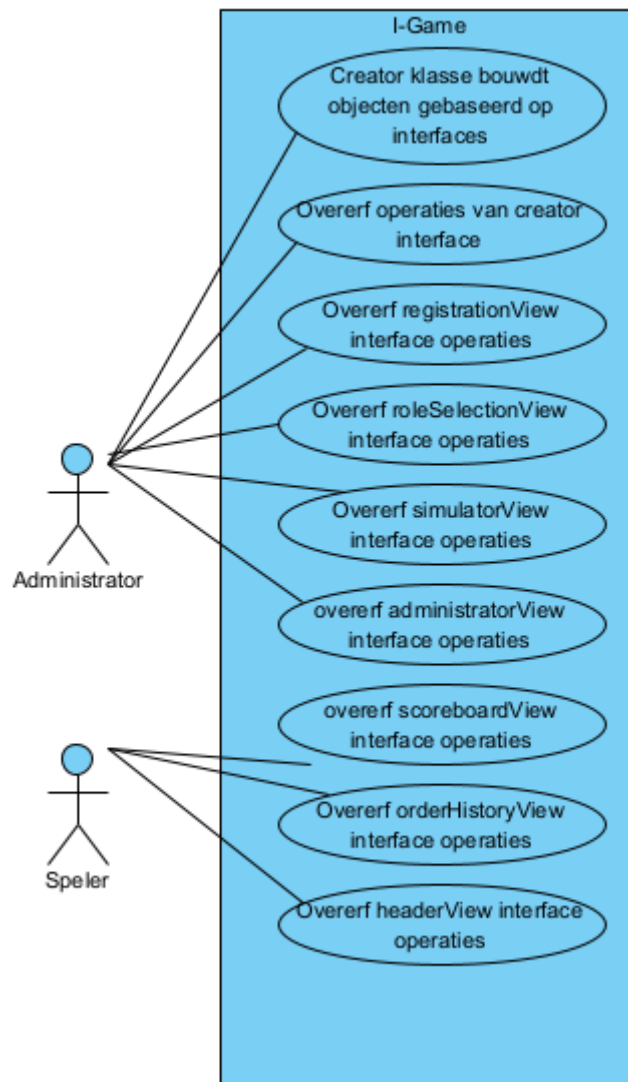
Usecase 8	Chain of Responsibility ontwerp patroon
Omschrijving	Het chain of responsibility patroon biedt een gebruikswijze om aanvragen tussen gebruikers te versturen, afhandelen of op te vangen.
Pre-condition	De applicatie en de simulatie zijn gestart
Post-condition	Alle aanvragen zijn opgevangen of afgehandeld.
Actoren	Speler
Main course of action	<ul style="list-style-type: none"> • US58: Een interface voor het afhandelen van aanvragen moet gemaakt worden. • US59: De interface moet door elke instantie van een speler aanroepbaar zijn. • US60: Elke instantie van een speler moet de operatie van de interface overnemen en voor zichzelf invullen (inheritance). • US61: Elke instantie van een speler moet een "Sell order" kunnen ontvangen. • US62: Elke instantie van een speler moet een "Buy order" kunnen ontvangen. • US63: Elke instantie die een aanvraag kan afhandelen controleert voor elke aanvraag of dit het geval is. Zo niet dan wordt de aanvraag doorgestuurd naar de volgende instantie die een aanvraag kan afhandelen. • US64: De interface moet een aanvraag die niet afgehandeld kan worden opvangen.
Exceptions	Wanneer een aanvraag niet kan worden afgehandeld door een van de afhandelaars dan wordt de aanvraag teruggestuurd naar de interface om door de handler interface afgehandeld te worden. Als een aanvraag niet afgehandeld kan worden dan wordt de aanvraag gesloten.



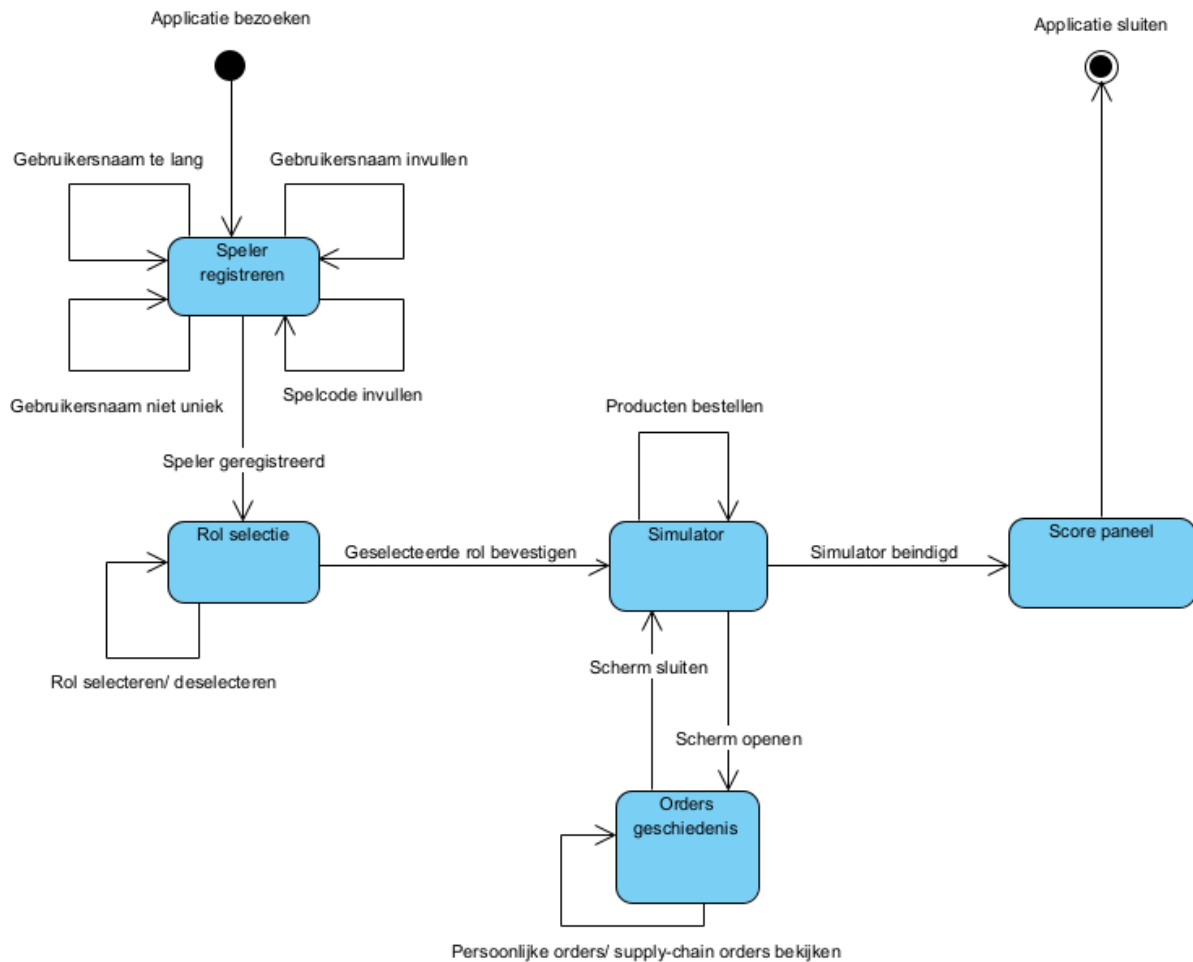
Figuur 22. Usecase diagram. Chain of responsibility

3.2.10 Usecase 10

Usecase 8	Factory Method ontwerp patroon
Omschrijving	Doormiddel van het fabriek patroon wordt er een gestructureerde en gecentraliseerde structuur vormgegeven voor het instantiëren van alle mogelijk te instantiëren klassen in de applicatie.
Pre-condition	De applicatie is gestart.
Post-condition	De administrator kan het administrator scherm aanroepen en de spelers kunnen het registreer scherm aanroepen.
Actoren	Administrator, Speler
Main course of action	<p>Wanneer een klasse geïnstantieerd moet worden wordt de fabriek aangeroepen die doormiddel van het overerven van operaties uit de fabriek interface klassen kan instantiëren en dan de objecten teruggeeft als referentie. Doormiddel van de referentie wordt het object aangeroepen.</p> <ul style="list-style-type: none"> • US65: Er moet een interface gemaakt worden voor de registrationView klasse. • US66: Er moet een interface gemaakt worden voor de roleSelectionView klasse. • US67: Er moet een interface gemaakt worden voor de simulatorView klasse. • US68: Er moet een interface gemaakt worden voor de administratorView klasse. • US69: Er moet een interface gemaakt worden voor de scoreboardView klasse. • US70: Er moet een interface gemaakt worden voor de orderHistoryView klasse. • US71: Er moet een interface gemaakt worden voor de HeaderView klasse. • US72: Een abstracte klasse moet vormgegevens worden met operaties voor het creëren van een objecten gebaseerd op de interfaces. • US73: Een creator klasse moet worden vormgegeven die operaties van de abstracte klasse kan implementeren en overschrijven als methoden. <p>In het usecase diagram op de volgende pagina worden twee actors weergegeven net zoals in deze usecase beschrijving. De twee actors worden weergegeven omdat de verschillende userstories indirect door een actie van de actors worden uitgevoerd.</p> <p>Bijvoorbeeld het order geschiedenis scherm kan alleen worden weergegeven als de speler op de hiervoor bestemde knop drukt.</p>
Exceptions	Het administrator scherm kan maar een keer geïnstantieerd worden.



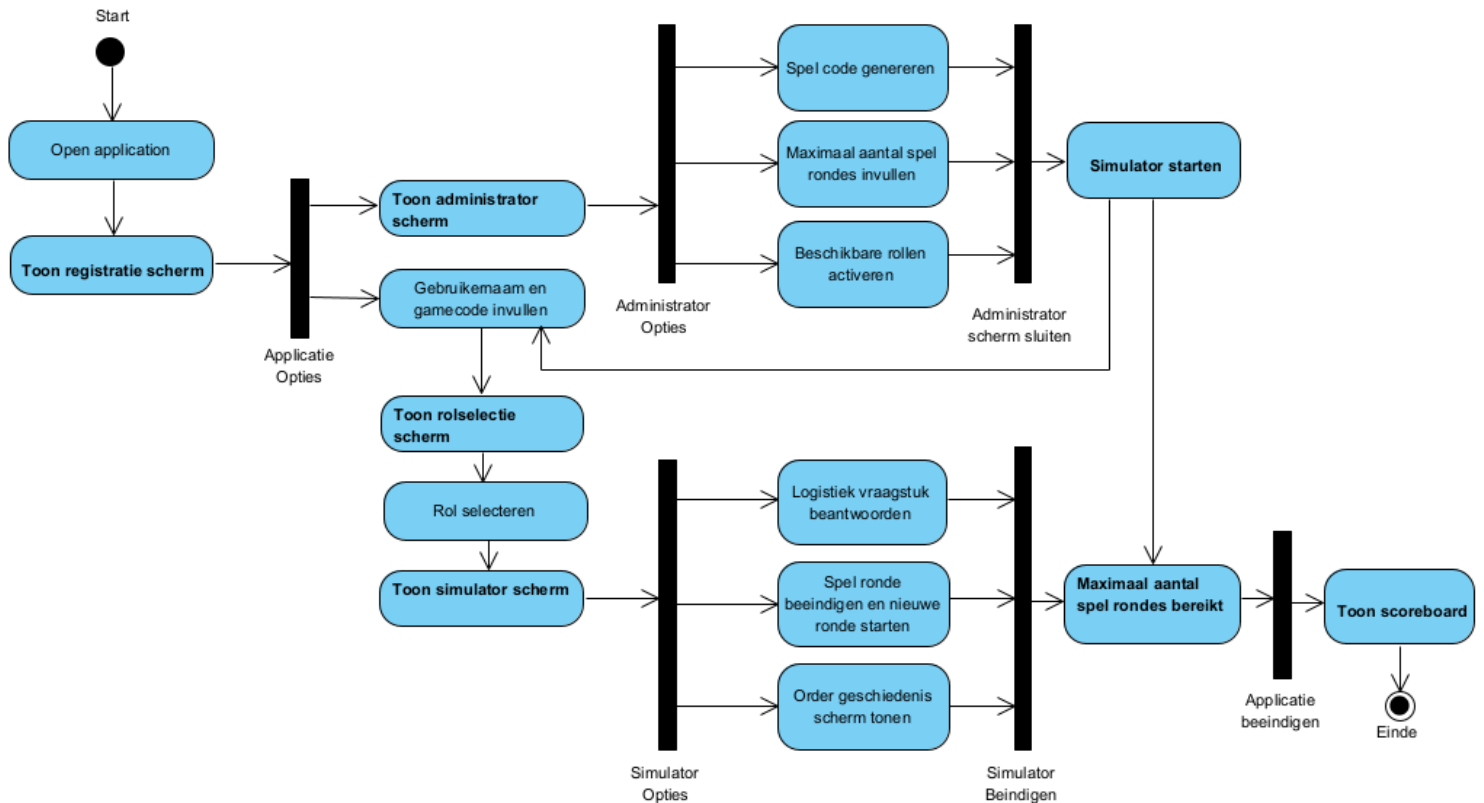
3.3 Toestandsdiagram



Figuur 8. Toestandsdiagram

In Figuur 8 wordt het toestand diagram (Statemachine diagram) getoond. Het doorloopt alle verschillende staten die de applicatie kan hebben en hoe hier doorheen gelopen wordt door de speler/ gebruiker van de applicatie.

3.4 Activity diagram



Figuur 9. Activity diagram

Hierboven wordt het activity diagram getoond. Dit diagram toont hoe de gebruiker door de applicatie heen loopt en welke activiteiten uitgevoerd kunnen worden.

De gebruiker krijgt het registratiescherm te zien bij het starten van de applicatie. Vanuit dit scherm kan het administrator scherm geopend worden of er kunnen een gebruikersnaam en gamecode worden ingevuld op het scherm.

Via het administrator scherm kan een gamecode worden gegenereerd, het aantal spel rondes worden ingevuld en de beschikbare rollen kunnen worden geselecteerd. De gegenereerde gamecode wordt getoond op het scherm. De administrator deelt deze code met de spelers.

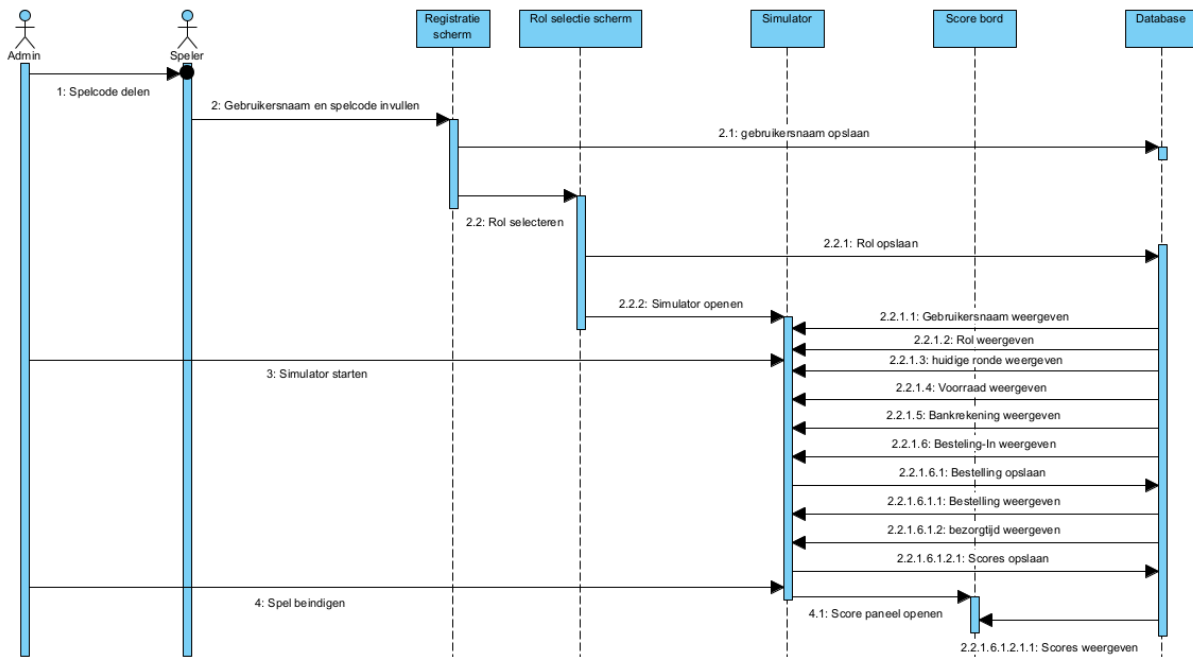
De spelers kunnen nu de gamecode invullen samen met een gebruikersnaam. Wanneer alle spelers de code en een naam hebben ingevuld en dit hebben bevestigd kan de administrator de simulator starten.

De spelers worden nu het rol selectie scherm getoond en kunnen een rol selecteren waarna de simulator wordt getoond.

In de simulator hebben de spelers de keuze uit meerdere opties. De spelers kunnen een order plaatsen, de order geschiedenis bekijken of de spel ronde beëindigen waarna een nieuwe ronde start. Het is belangrijk dat de speler de getoonde data goed bekijkt en gebaseerd daarop bepaald of de speler meer of minder producten nodig heeft.

Wanneer het maximaal aantal spel rondes is bereikt wordt de simulator beëindigt en wordt het scoreboard getoond waar de getoonde data een kort overzicht geeft van de speler.

3.5 Sequence diagram



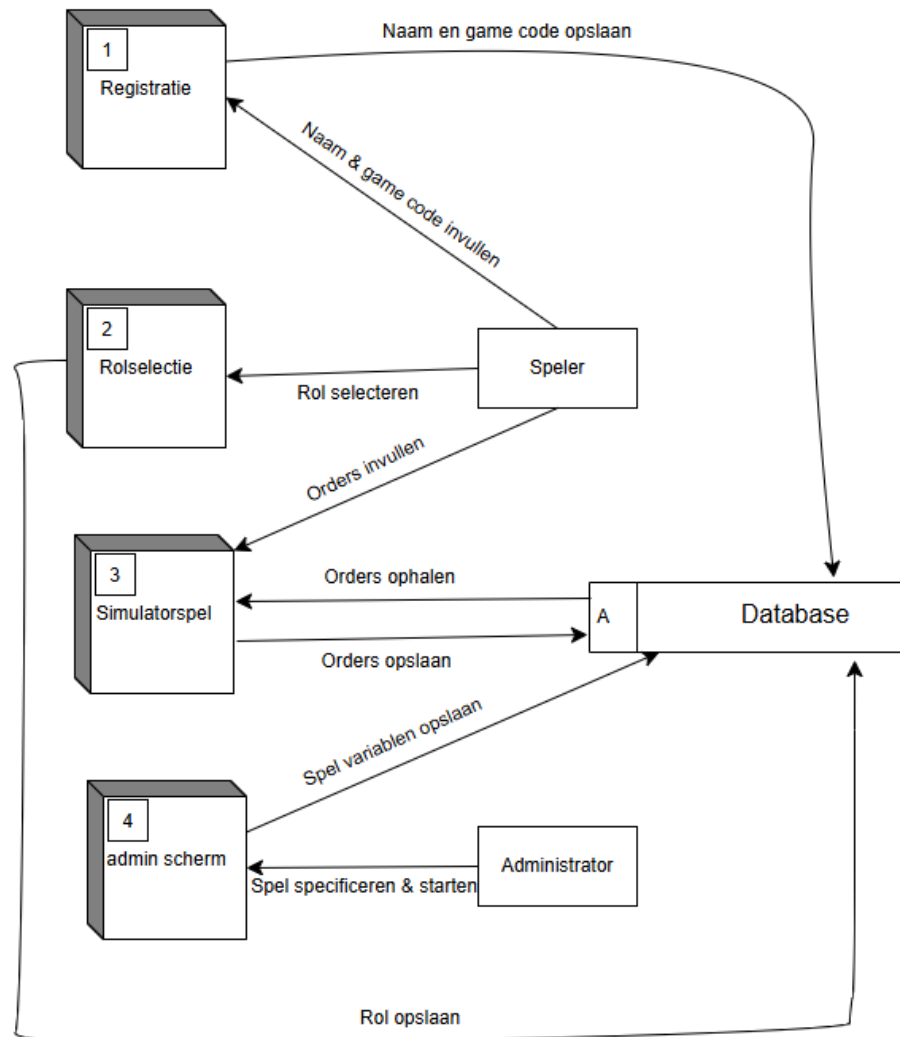
Figuur 10. Sequence diagram

In figuur 10 wordt het Sequence diagram getoond. Het sequence diagram toont welke elementen actief zijn over de loop van de uitgevoerde applicatie.

Ronduit de meeste functionaliteiten en acties zitten in het simulator spel waarin de applicatie meermaals data moet ophalen uit de database maar ook andere data moet laten versturen naar de database.

De andere pagina`s versturen alleen data naar de database.

3.6 Gegevensstromen



Figuur 11. Data flow diagram

In Figuur 11 worden de gegevens stromen van het simulator spel weergegeven.

Zoals te zien bestaat er een database en moet de speler eerst een naam registreren en rol selecteren voordat de speler het simulatorspel kan spelen.

De administrator heeft de mogelijkheid om het simulator spel aan te passen aan de vereiste variabelen voor het spel. Ook beslist de administrator wanneer het spel gestart wordt.

Het data flow diagram is op level 1 opgesteld met behulp van de volgende documentatie:

<https://www.visual-paradigm.com/guide/data-flow-diagram/what-is-data-flow-diagram/>

3.7 Schermontwerpen

Gebaseerd op de diagrammen in eerdere hoofdstukken en de user story's zoals opgesteld in hoofdstuk 2.2 moeten er schermontwerpen worden opgesteld. Voor de schermontwerpen worden de SOLID principes gebruikt zoals beschreven in hoofdstuk 1.2

3.7.1 Login scherm

The wireframe shows a browser window titled 'Login page'. Inside, there's a header bar with the text 'Intusion game' on the left and an 'Admin' button on the right. The main content area is centered and contains the following elements from top to bottom: a label 'Gebruikersnaam' above a text input field containing 'Gebruikersnaam'; a label 'Spelcode' above a text input field containing '1234'; a 'Registreren' button; and a red error message box with the text 'Naam is mogelijk al in gebruik. Type een nieuwe naam'.

Figuur 12. Schermontwerp van de login pagina

In figuur 12 geeft het login scherm weer. De gebruikers van de applicatie zullen via dit scherm hun gebruikersnaam invullen evenals de spelcode die door de spel administrator wordt uitgedeeld aan de spelers van het spel.

De speler kan een gebruikersnaam zelf bedenken en in het tekst vak typen. De gebruikersnaam mag niet langer zijn dan twintig tekens en moet uniek zijn.

Wanneer de speler op de registreren knop drukt zal de gebruikersnaam worden opgeslagen en wordt de speler naar het rol selectie scherm doorgestuurd.

Via dit scherm is er ook de mogelijkheid voor de administrator om het admin paneel te openen waar de admin het spel kan beheren.

Op de volgende pagina worden de userstories en bijkomende acceptatiecriteria weergegeven die gebruikt zijn voor het ontwerpen van dit schermontwerp.

- **US01:** Als speler wil ik mijn gebruikersnaam kunnen invullen in een tekst vak, zodat de prestaties die ik behaal aan mij gekoppeld kunnen worden.
 - Er moet een label met de tekst “Gebruikersnaam” worden weergegeven.
 - Er moet een tekst vak met een placeholder worden getoond.
 - De placeholder van het tekst vak bevat de tekst “Gebruikersnaam”
- **US02:** Als speler wil ik op de registreren knop kunnen drukken, zodat mijn ingevulde gebruikersnaam wordt opgeslagen.
 - Er moet een knop met de tekst “Registreren” worden getoond.
 - Er moet een label met de tekst “Spel code” worden getoond.
 - Er moet een tekst vak worden weergegeven waarin de speler de spel code kan typen.
- **US03:** Als speler wil ik een rood veld getoond krijgen, zodat ik kan zien dat mijn gekozen gebruikersnaam al in gebruik is.
 - Er wordt een roodgekleurd label weergegeven als het tekst vak is ingevuld met een gebruikersnaam die al geregistreerd is en op de registreren knop is geklikt.
 - Het roodgekleurde label bevat de tekst “Gebruikersnaam is mogelijk al in gebruik. Type een nieuwe gebruikersnaam”.
- **US04:** Als speler wil ik een rood veld getoond krijgen, zodat ik kan zien dat mijn gebruikersnaam te lang is.
 - Er wordt een roodgekleurd label weergegeven als het tekst vak is ingevuld met een gebruikersnaam die meer dan twintig tekens bevat en de registreren knop is geklikt.
 - Het roodgekleurde label bevat de tekst “Gebruikersnaam is te lang. Type een kortere gebruikersnaam”.
- **US05:** Als speler word ik na het succesvol registreren doorgestuurd naar het rolselectie scherm, zodat ik een rol kan selecteren.
 - Nadat de speler een gebruikersnaam heeft ingevuld en op de registreren knop heeft gedrukt en de juiste spel code heeft ingevuld wordt de input gecontroleerd speciale tekens (Validatie).
 - Als validatie voltooid is wordt de gebruikersnaam opgeslagen.
 - De spel code wordt gecontroleerd.
 - De speler wordt na succesvol registreren doorgestuurd naar het rol selectie scherm.

3.7.2 Rol selectie scherm

Rol Selectie

Intuition game Gebruikersnaam

Kies een rol voor het spel. Elke rol heeft zijn eigen te verwerken en leveren producten

Factory

Distributor

Wholesaler

Retailer

Bevestigen

Figuur 13. Schermontwerp van de rol selectie pagina

In figuur 13 wordt het rol selectie scherm weergegeven. Via dit scherm kan de speler een rol selecteren voor het simulator spel. Als de speler een rol selecteert zal de tekst op de knop dikgedrukt worden weergegeven. Door op de bevestigen knop te drukken wordt de keuze van de speler opgeslagen en wordt het de speler doorgestuurd naar het simulator spel.

Boven aan de pagina ziet de speler de gebruikersnaam zoals zij deze geregistreerd hebben.

Op de volgende pagina worden de userstories en bijkomende acceptatiecriteria weergegeven die gebruikt zijn voor het ontwerpen van dit schermontwerp.

- **US16:** Als speler wil ik minimaal uit vier bedrijf takken kunnen kiezen, zodat ik verschillende rollen kan spelen.
 - Er moet een label met de tekst “Kies een rol voor het spel. Elke rol heeft zichzelf te verwerken en te leveren producten”.
 - Voor elke rol wordt een uitleg tekst getoond in een label onder de opties.
- **US17:** Als speler wil ik mijn keuze kunnen bevestigen, zodat ik niet de verkeerde rol selecteer.
 - Er moet een knop met de tekst “Bevestigen” weergegeven worden.
- **US18:** Als speler wil ik naar het simulator spel kunnen navigeren, zodat ik elk onderdeel binnen de simulatie makkelijk kan bereiken.
 - Door het klikken op de bevestiging knop wordt de rol keuze geregistreerd en opgeslagen
 - De speler wordt doorgestuurd naar het simulator spel.
- **US38:** Als speler wil ik de rol van Factory kunnen kiezen, zodat ik de producten kan ontwikkelen.
 - Er moet een knop met de tekst “Factory” getoond worden.
 - De kleur van de knop verandert als deze wordt geklikt.
- **US39:** Als speler wil ik de rol van Distributor kunnen kiezen, zodat ik de producten kan verzenden.
 - Er moet een knop met de tekst “Distributor” getoond worden.
 - De kleur van de knop verandert als deze wordt geklikt.
- **US40:** Als speler wil ik de rol van Wholesaler kunnen kiezen, zodat ik de producten grootschalig kan verkopen.
 - Er moet een knop met de tekst “Wholesaler” getoond worden.
 - De kleur van de knop verandert als deze wordt geklikt.
- **US41:** Als speler wil ik de rol van Retailer kunnen kiezen, zodat ik producten kan kleinschalig kan verkopen.
 - Er moet een knop met de tekst “Retailer” getoond worden.
 - De kleur van de knop verandert als deze wordt geklikt.

3.7.3 Simulator scherm

The screenshot shows a window titled "Simulator (Versie B)". Inside, there's a header bar with "Intusion game" on the left and "Gebruikersnaam" on the right. Below this, the main content area is divided into several sections. At the top, there are two tabs: "Wholesaler" (selected) and "Dag 1". The main area contains six boxes arranged in a 2x3 grid: "Bestellingen" (150 L), "Ontvangen" (0000 L), "Geleverd" (50 L) in the top row, and "Voorraad" (50 L), "Levertijd" (1 Dag), "Backorder" (100 L) in the bottom row. At the bottom of the window, there are three sections: "Bestelling plaatsen" with a text input "100 L" and a "Confirm" button; "Bank Rekening" with a text input "€ 1.000"; and "Orders" with an empty square button.

Figuur 14. Schermontwerp van de simulator pagina

In figuur 14 wordt in het schermontwerp het simulator spel weergegeven zoals de speler deze zal zien.

De speler heeft via dit scherm een overzicht van hun bedrijf. De speler ziet hoeveel bestellingen andere bedrijven bij hen een bestelling hebben geplaatst. Onder de bestellingen wordt de beschikbaar bedrijf voorraad getoond. Als de speler nog voldoende voorraad heeft zal deze voorraad worden gebruikt om binnen gekomen bestellingen te leveren. Als er geen voorraad beschikbaar is en de speler voorraad moet bij bestellen zal de bestelling worden omgezet naar een backorder en zal de bestelling pas geleverd worden als er nieuwe producten zijn ontvangen.

De speler kan nieuwe producten bestellen door in het tekst vak onderaan de pagina het benodigde aantal producten in te voeren en daarna op bevestigen te klikken. De levertijd van de bestelde producten wordt getoond in het vak met als titel levertijd.

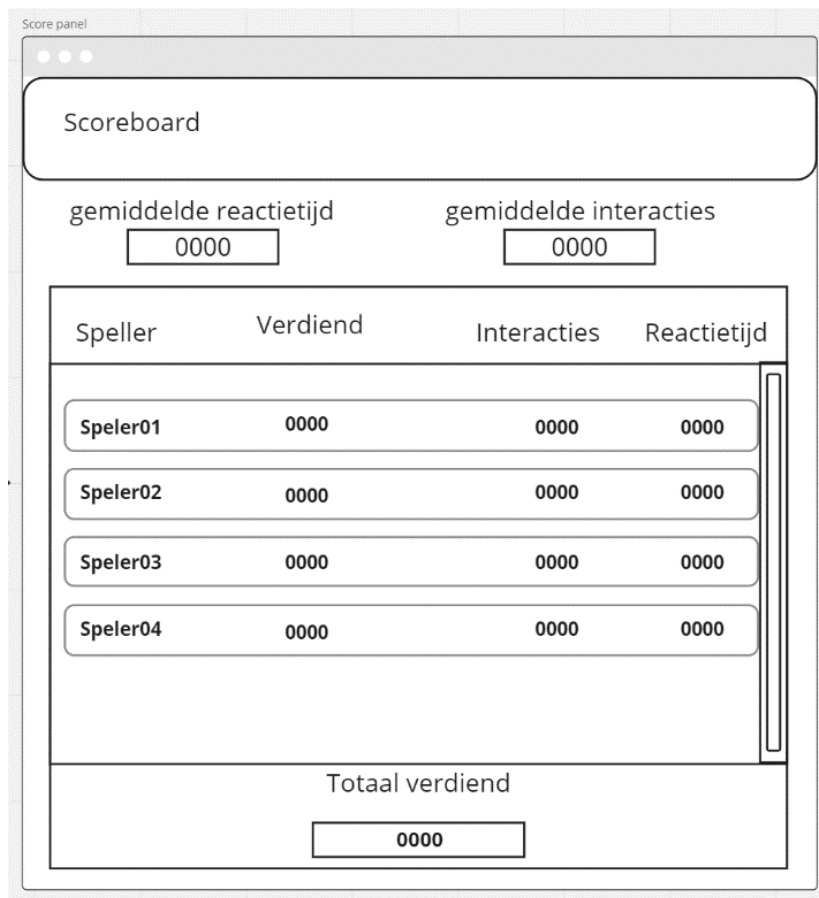
Verder heeft de speler een bankrekening. De speler ontvangt een geldbedrag door succesvol een bestelling te leveren en verliest geld door meer producten te bestellen. De speler kan de afgeronde bestellingen bekijken voor zowel zichzelf als voor de gehele supplychain door op de knop Orders te klikken.

Op de volgende pagina worden de userstories en bijkomende acceptatiecriteria weergegeven die gebruikt zijn voor het ontwerpen van dit schermontwerp.

- **US29:** Als speler wil ik aan het begin zien welke rol ik heb in het spel, zodat ik weet welke rol, aldus welke taak ik verantwoordelijk voor ben.
 - Er moet een tekst vak met de naam van rol worden weergegeven.
- **US30:** Als speler wil ik mijn gebruikersnaam kunnen zien tijdens het spelen van het spel, zodat ik weet met welke naam ik het spel speel.
 - Er moet een label met de naam van de speler getoond worden op het simulator scherm.
- **US31:** Als speler wil ik zien hoeveel klanten een bestelling hebben aangemaakt, zodat ik weet hoeveel vracht ik nodig ga hebben.
 - Er moet een vierkant vak getoond worden
 - Er moet een label met de tekst “Bestellingen” getoond worden in het vak
 - Er moet een label met een cijfer getoond worden in het vak.
 - Er moet een eenheid in liters “L” gebruikt worden om het aantal bestellingen aan te duiden.
- **US32:** Als speler wil ik zien hoeveel voorraad mijn bedrijf heeft, zodat ik weet hoeveel producten ik moet bestellen.
 - Er moet een vierkant vak getoond worden
 - Er moet een label met de tekst “Voorraad” weergegeven worden in het vak.
 - Er moet een label met een cijfer weergegeven worden in het vak.
 - Er moet een eenheid in liters “L” gebruik worden om de het aantal voorraad aan te duiden i het vak.
- **US33:** Als speler wil ik producten kunnen bestellen, zodat ik deze kan verwerken en versturen naar mijn klanten.
 - Er moet een label met de tekst “Bestelling plaatsen” getoond worden in het vak.
 - Er moet een tekst vak waarin getypt kan worden getoond worden in het vak.
 - Er moet een eenheid in liters gebruikt worden om het aantal te bestellen producten aan te duiden in het vak.
 - Er moet een knop met de tekst “Bevestig” weergegeven worden in het vak.
- **US34:** Als speler wil ik zien hoeveel levertijd er is voor de producten die ik bestel, zodat ik weet hoelang het duurt voordat mijn bestelling aankomt.
 - Er moet een vierkant vak getoond worden.
 - Er moet een label met de tekst “Levertijd” weergegeven worden in het vak.
 - Er moet een label met een cijfer getoond worden in het vak.
 - Er moet een eenheid in dagen/dag gebruikt worden om de levertijd aan te duiden.
- **US35:** Als speler wil ik zien hoeveel bestellingen op pauze staan, zodat ik weet welke bestellingen ik vroeg of laat moet voortzetten.
 - Er moet een vierkant vak getoond worden.
 - Er moet een label met de tekst “Backorder” getoond worden in het vak.
 - Er moet een label met een cijfer getoond worden in het vak.
 - Er moet een eenheid in liters “L” gebruikt worden om het aantal backorders aan te duiden in het vak.
- **US36:** Als speler wil ik zien hoeveel spel rondes er momenteel zijn afgerond, zodat ik weet hoever ik ben met het spel.
 - Er moet en tekst vak met een cijfer getoond worden.
 - Er moet een eenheid in dagen/dag gebruikt worden om het aantal rondes aan te duiden.

- **US37:** Als speler wil ik zien hoeveel producten er aan mij geleverd worden, zodat ik weet hoeveel producten ik ontvang.
 - Er moet een vierkant vak getoond worden.
 - Er moet een label met de tekst “Ontvangen” weergegeven worden in het vak.
 - Er moet een label met een cijfer weergegeven worden in het vak.
 - Er moet een eenheid in liters “L” gebruikt worden om het aantal ontvangen producten in de huidige ronde aan te duiden.
- **US44:** Als speler wil ik zien hoeveel geld ik bezit, zodat ik weet hoeveel producten ik kan bestellen.
 - Er moet een label met de tekst “Bankrekening” weergegeven worden.
 - Er moet een tekst vak met een bedrag getoond worden.
 - Er moet een eenheid in euro’s “€” gebruikt worden om de hoeveelheid geld aan te duiden dat beschikbaar is voor de speler.
- **US45:** Als speler wil ik een bedrag ontvangen als ik een product geleverd heb, zodat ik een beloning krijg voor mijn uitgevoerde werk.
 - Per geleverd product moet een bedrag toegevoegd worden aan de bankrekening.
- **US46:** Als speler wil ik een bedrag betalen als ik een product bestel bij een ander bedrijf, zodat deze tak een reden heeft om het product aan mij te leveren.
 - Per besteld product wordt een bedrag van de bankrekening afgetrokken als de speler op bevestig klikt.

3.7.4 Score panel



The mockup shows a window titled 'Score panel' containing a 'Scoreboard' section. At the top, there are two summary metrics: 'gemiddelde reactietijd' (average reaction time) and 'gemiddelde interacties' (average interactions), both with input fields showing '0000'. Below these is a table with four columns: 'Speller', 'Verdiend', 'Interacties', and 'Reactietijd'. The table lists four players (Speler01 to Speler04), each with corresponding '0000' values in the other three columns. A vertical scrollbar is on the right side of the table. At the bottom of the table, there is a 'Totaal verdiend' (Total earned) label and an input field showing '0000'.

Speller	Verdiend	Interacties	Reactietijd
Speler01	0000	0000	0000
Speler02	0000	0000	0000
Speler03	0000	0000	0000
Speler04	0000	0000	0000
Totaal verdiend		0000	

Figuur 15. Schermontwerp van het scorepaneel

In figuur 15 is het schermontwerp voor het scorepaneel getoond. De speler zal via dit scherm kunnen na gaan hoe effectief ze waren specifiek hoeveel geld ze verdiend hebben. De reactietijd en het aantal interacties worden ook weergegeven. Ook wordt getoond hoe effectief de gehele supplychain is geweest door het totaal verdiende bedrag van de supplychain te tonen samen met de gemiddelde reactietijd en gemiddeld aantal interacties.

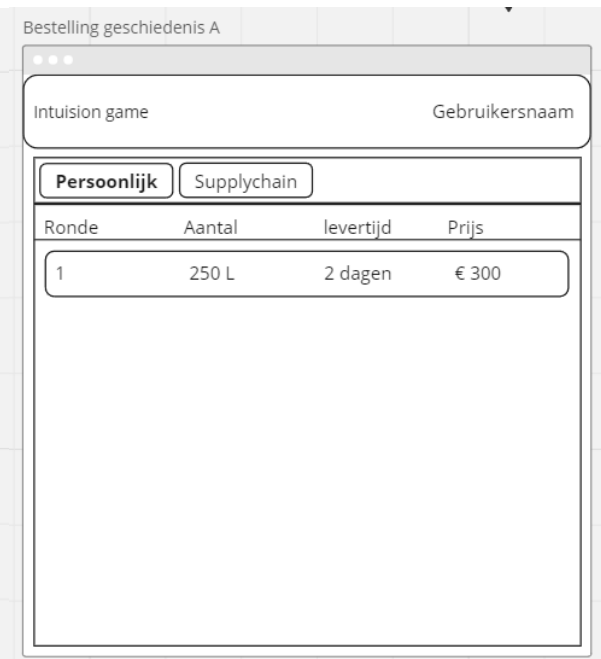
Op de volgende pagina worden de userstories en bijkomende acceptatiecriteria weergegeven die gebruikt zijn voor het ontwerpen van dit schermontwerp.

- **US19:** Als speler wil ik op het einde van een spel mijn reactietijd kunnen inzien, zodat ik kan zien hoe snel ik mijn beslissingen heb genomen.
 - Er moet een label met de tekst “reactietijd” weergegeven worden.
 - Er moet een tekst vak met een cijfer worden weergegeven.
 - Er moet seconden als eenheid gebruikt worden.
- **US20:** Als speler wil ik op het einde van een spel mijn interacties kunnen inzien, zodat ik weet wat voor beslissingen ik heb genomen tijdens het spel.
 - Er moet een label met de tekst “interacties” weergegeven worden.
 - Er moet een tekst vak met een cijfer worden weergegeven.
- **US21:** Als speler wil ik een scoreboard zien aan het einde van het spel, zodat ik kan zien wie het beste heeft gespeeld.
 - Er moet een record weergegeven worden per speler.
 - Er moet een label met de tekst “Speler” weergegeven worden.
 - Er moet een gebruikersnaam per record worden weergegeven.
 - Er moet een label met de tekst “verdiend” worden weergegeven.
 - Er moet een label met de tekst “interacties” worden weergegeven.
 - Er moet een label met de tekst “reactietijd” worden weergegeven.
- **US22:** Als speler wil ik een gemiddelde reactietijd zien aan het einde van het spel, zodat ik kan zien hoe snel de gemiddelde beslissing werd genomen.
 - Er moet een label met de tekst “gemiddelde reactietijd” worden weergegeven.
 - Er moet een tekst vak met een cijfer weergegeven worden.
- **US23:** Als speler wil ik een gemiddelde aantal interactie zien aan het einde van het spel, zodat ik kan zien hoeveel acties er werden genomen in het spel.
 - Er moet een label met de tekst “gemiddelde interacties” worden weergegeven.
 - Er moet een tekst vak met een cijfer weergegeven worden.
- **US43:** Als speler wil ik een bedrag zien dat ik heb verdiend doormiddel van het produceren, verwerken en leveren van producten, zodat ik kan zien hoeveel profijt/verlies ik heb gekregen.
 - Er moet een label met de tekst “verdiend” worden weergegeven.
 - Er moet een tekst vak met een cijfer weergegeven worden.
 - Er moet een eenheid in euro’s “€” gebruikt worden.
- **US51:** Als speler wil ik zien hoeveel de gehele supplychain aan bedrag heeft verdiend, zodat ik een inschatting van mijn prestaties heb.
 - Er moet een label met de tekst “Totaal verdiend” worden weergegeven
 - Er moet een tekst vak onder het label worden weergegeven
 - Er moet een cijfer worden getoond in het tekst vak
 - Er moet een euro symbool worden getoond in het tekst vak als eenheid.
- **US52:** Als speler wil ik door de lijst met spelers kunnen scrollen, zodat ik kan zien wie er allemaal meedoet.
 - Er moet een scrol balk worden getoond
 - De scrol balk moet naar beneden en naar boven kunnen bewegen

3.7.5 Bestelling geschiedenis



Figuur 16. Schermontwerp van de bestel geschiedenis pagina voor de supplychain



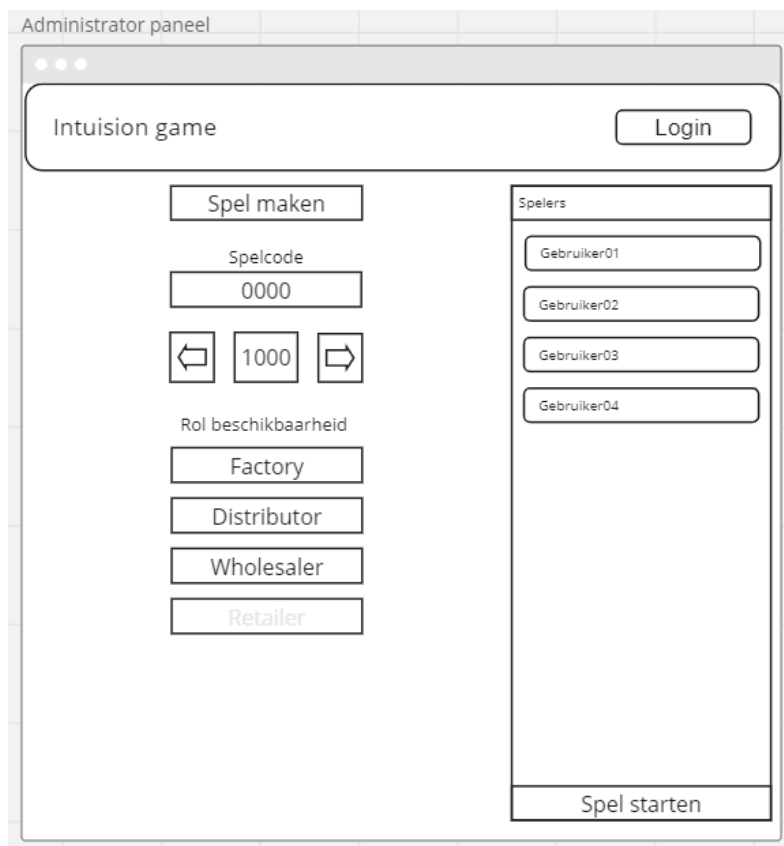
Figuur 17. Schermontwerp van de bestel geschiedenis pagina voor de supplychain

In figuur 16 en 17 zijn de schermontwerpen voor het scorepaneel getoond. De speler kan doormiddel van dit scherm de bestellingen zien die bij zichzelf zijn binnengekomen en zijn verzonden. Ook is er de mogelijkheid om de bestellingen van de gehele supplychain te bekijken. Hiermee wordt bedoeld de orders die het laatste bedrijf in de supplychain heeft aangemaakt en afgerond.

Op de volgende pagina worden de userstories en bijkomende acceptatiecriteria weergegeven die gebruikt zijn voor het ontwerpen van dit schermontwerp.

- **US47:** Als speler wil ik een scherm kunnen openen tijdens de simulator, zodat ik mijn bestellingen kan zien.
 - Er moet een scherm geopend worden als de speler op de orders knop drukt op het simulator scherm.
 - Het scherm wordt geopend op de voorgrond.
 - Het scherm kan geopend zonder de simulator te pauzeren.
- **US48:** Als speler wil ik het verdiende bedrag, de levertijd, aantal en ronde per afgeronde bestelling zien, zodat ik weet hoeveel ik verdiend heb.
 - Er moet per afgeronde bestelling een record getoond worden.
 - Er moet een label met de tekst "Ronde" weergegeven worden.
 - Er moet een label met een cijfer voor het ronde nummer getoond worden.
 - Er moet een label met de tekst "Aantal" weergegeven worden.
 - Er moet een label met een cijfer voor het aantal bestelde producten getoond worden.
 - Er moet een eenheid in liters "L" worden getoond voor het product aantal.
 - Er moet een label met de tekst "Levertijd" weergegeven worden.
 - Er moet een label met een cijfer voor de levertijd getoond worden.
 - Er moet een eenheid in "dagen" worden getoond voor de levertijd.
 - Er moet een label met de tekst "Prijs" weergegeven worden
 - Er moet een label met een cijfer voor prijs bedrag getoond worden
 - Er moet een eenheid in euro's "€" getoond worden voor de prijs
- **US49:** Als speler wil ik een scherm kunnen openen tijdens de simulator, zodat ik de bestellingen van de gehele supplychain kan zien.
 - Er moet een knop met de tekst "Persoonlijk" weergegeven worden.
 - Er moet een knop met de tekst "Supplychain" weergegeven worden.
 - De tekst in een van de twee knoppen is dikgedrukt (bold) als het scherm actief is.
 - Er moet per afgeronde bestelling een record getoond worden.
 - Er moet een label met de tekst "Ronde" weergegeven worden.
 - Er moet een label met een cijfer voor het ronde nummer getoond worden.
 - Er moet een label met de tekst "Aantal" weergegeven worden.
 - Er moet een label met een cijfer voor het aantal bestelde producten getoond worden.
 - Er moet een eenheid in liters "L" worden getoond voor het product aantal.
 - Er moet een label met de tekst "Levertijd" weergegeven worden.
 - Er moet een label met een cijfer voor de levertijd getoond worden.
 - Er moet een eenheid in "dagen" worden getoond voor de levertijd.
 - Er moet een label met de tekst "Prijs" weergegeven worden
 - Er moet een label met een cijfer voor prijs bedrag getoond worden
 - Er moet een eenheid in euro's "€" getoond worden voor de prijs
- **US50:** Als speler wil ik het totaal verdiende bedrag van een bestelling van de gehele supplychain zien, zodat ik kan inschatten hoe efficiënt de gehele supplychain is.
 - Er moet een label met de tekst "Totaal" weergegeven worden onder de bestelling geschiedenis van de gehele supplychain
 - Er moet een tekst vak met een bedrag en eenheid in euro's "€" getoond worden

3.7.6 Administrator functies



Figuur 18. Schermontwerp van de administrator pagina

In figuur 18 is het schermontwerp getoond voor het administratie paneel. Via dit scherm kan een administrator een spel aanmaken en bewerken. Een administrator kan het aantal rondes aangeven en specifieke rollen in het spel aan of uit zetten. Rechts van dit scherm zal een lijst met spelers worden weergegeven. Een speler komt in de lijst te staan wanneer zij een rol hebben gekozen en het simulator spel scherm hebben geopende.

Rollen die de administrator heeft uitgezet krijgen een licht grijze tekst op hun knop.

Op de volgende pagina worden de userstories en bijkomende acceptatiecriteria weergegeven die gebruikt zijn voor het ontwerpen van dit schermontwerp.

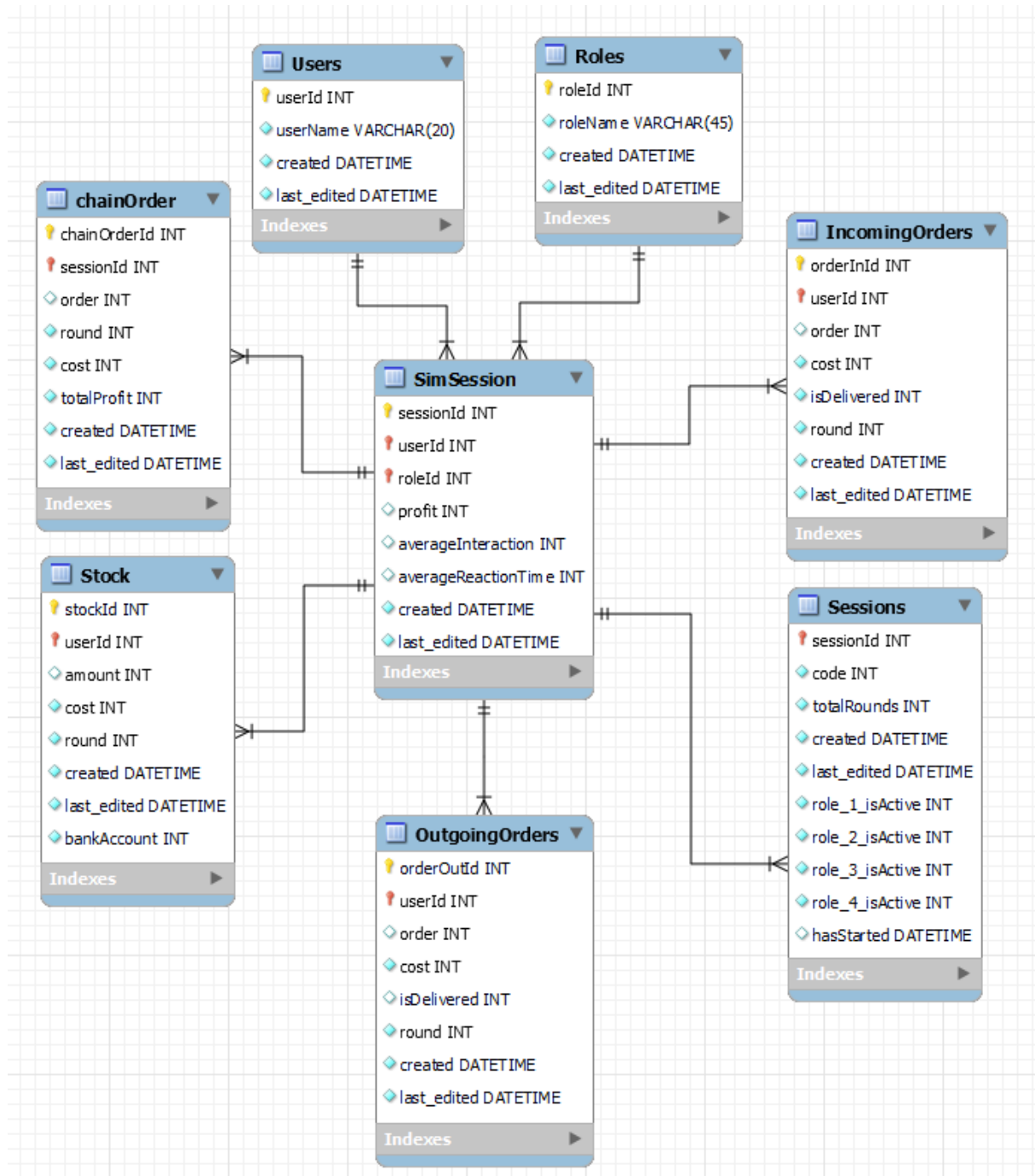
- **US24:** Als administrator wil ik een spel kunnen aanmaken, zodat de spelers toegang krijgen tot de simulatie.
 - Er moet een knop met de tekst “Spel maken” worden weergegeven.
- **US25:** Als administrator wil ik een spel kunnen starten, zodat de spelers het spel kunnen beginnen.
 - Er moet een knop met de tekst “Spel starten” worden weergegeven.
- **US26:** Als administrator wil ik een spel code kunnen zien, zodat de spelers weten welk spel ze moeten betreden.
 - Er moet een tekst vak met een cijfercode weergegeven worden.
 - De cijfercode wordt elke keer dat een spel wordt aangemaakt random gegenereerd.
 - De spel code bevat vier cijfers.
- **US27:** Als administrator wil ik het aantal spel rondes kunnen instellen, zodat ik kan bepalen hoelang het spel duurt.
 - Er moet een tekst vak met een cijfer aan aantal rondes weergegeven worden.
 - Er moet een knop voor het verhogen van het aantal rondes weergegeven worden
 - Er moet een knop voor het verlagen van het aantal rondes weergegeven worden.
- **US28:** Als administrator wil ik instellen welke bedrijfstakken de spelers kunnen uitkiezen, zodat ik kan kiezen welke partijen op het moment beschikbaar zijn.
 - Er moet een label met de tekst “Rol beschikbaar” getoond worden.
 - Er moet een knop met de tekst “Factory” getoond worden.
 - Er moet een knop met de tekst “Distributor” getoond worden.
 - Er moet een knop met de tekst “Wholesaler” getoond worden.
 - Er moet een knop met de tekst “retailer” getoond worden.
 - Als de admin op een actieve rol klikt veranderd de tekstkleur naar licht grijs
 - Als de admin op een non-actieve rol klik veranderd de tekstkleur naar zwart
- **US42:** Als administrator wil ik een lijst met de spelers kunnen zien voordat het spel begint, zodat ik weet welke spelers en hoeveel spelers er zijn.
 - Er moet een vak in de vorm van een rechthoek getoond worden.
 - Er moet voor elke speler dat een rol gekozen heeft een balk getoond worden in het vak.
 - De gebruikersnaam wordt in een label op de balk getoond.

4 Technische ontwerpen

4.1 ERD Database ontwerp

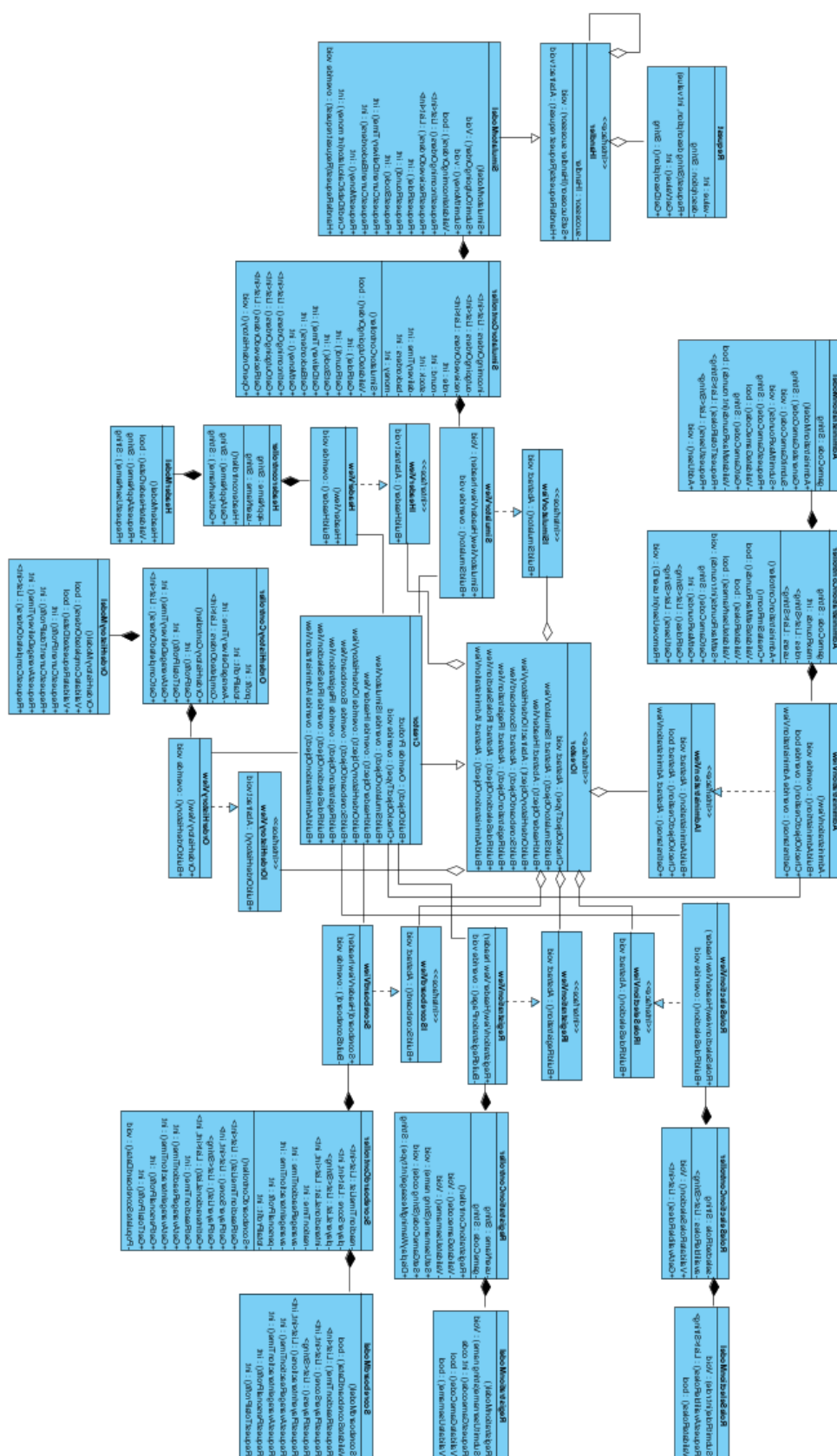
Het ERD zal op een later moment worden ontworpen als voor de schermontwerpen de uiteindelijke laatste versie is vastgesteld.

Het onderstaande ERD is de laatste versie van het ERD.



Figuur 19. ERD van de applicatie database

4.2 Class diagram



Figuur 20. Class diagram op gespecificeerd niveau. Version 3.5

In figuur 20 wordt het class diagram op gespecificeerd niveau getoond. Dit class diagram dient als concept en zal bij het realiseren worden uitgebreid om de volledige laravel applicatie functionaliteiten te tonen. Momenteel toont het class diagram de vereiste klassen en hun elementen om alle usecases in specifieke concept te tonen.

De applicatie zoals in figuur 19 getoond wordt is in drie delen opgedeeld volgens het MVC-concept, Model, View en Controller.

In het MVC-model gecombineerd met OOP wordt voor elke view een controller en een model aangemaakt. De view behandelt alle functionaliteiten die direct het scherm tekenen zoals in de klasse wordt beschreven. De controller klasse behandelt alle indirecte functionaliteiten voor de view en tegelijkertijd is de controller ook de klasse die communiceert met het model. Het model behandelt alle functionaliteiten die met de database en data validatie te maken hebben.

Door OOP met MVC te combineren wordt er aan de controller meer functionaliteiten toegewezen. De controller krijgt hierdoor de taak om de objecten van de view en model aan te roepen en te creëren. Op deze manier zorgt de controller ervoor dat het model en view altijd op de juiste manier en tijd worden aangeroepen. Dit betekent dus ook dat wanneer er een view aangeroepen moet worden omdat bijvoorbeeld een nieuw scherm getoond moet worden dat de view niet aangeroepen zal worden maar de controller. Wanneer de controller wordt aangeroepen zal de controller eerst het model instantiëren en aanroepen. Hierna wordt de view geïntanceerd en aangeroepen waarop de gebruiker de view te zien zal krijgen en zal de data die het model opgevraagd en gevalideerd heeft uit de database getoond worden in de view.

In versie 3.0+ van dit project zijn er drie ontwerp patronen toegevoegd aan het ontwerpdocument. Het eerste ontwerp patroon is het *Singleton patroon* dat voor een of meerdere klassen een gebruikswijze biedt om een klasse altijd maar een object te laten hebben en dit kan controleren.

Het tweede ontwerp patroon is het *Factory method patroon* dat een gebruikswijze beid voor het standaardiseren en gestructureerd instantiëren van objecten binnen een applicatie.

Het derde ontwerp patroon is het *Chain of responsibility patroon* dat een gebruikswijze biedt voor het gestructureerd en gestandaardiseerd versturen en afhandelen van aanvragen binnen een applicatie.

Wanneer de applicatie wordt gestart moet als eerst het registratie scherm getoond worden. Het registratie scherm wordt aangeroepen door de registrationController die na het instantiëren en aanroepen van het registrationModel de registrationView instantieert en aanroept. Het registrationModel haalt de data uit de database en stuurt deze na het valideren door naar de registrationController.

De registrationController kan de registrationModel direct instantieeren en aanroepen maar voor de registratorView wordt een *factory* gebruikt om de klasse te instantiëren. Dit werkt op de volgende manier: De registrationController stuurt een verzoek voor het instantiëren van de registrationView naar de Creator klasse die methoden vanuit de ICreator interface heeft geïmplementeerd voor het creëren van objecten.

De Creator klasse instantieert dan de te instantiëren klasse en stuurt deze terug naar de registrationController doormiddel van het return type van de BuildObject methode. Nu kan de registrationController de registrationView aanroepen zodat het scherm wordt getoond aan de gebruiker van de applicatie.

De speler van de applicatie kan alleen verder als zij de game code van de simulatie hebben ontvangen van de administrator. De administrator kan het administrator scherm openen door een knop in het registratie scherm(registrationView). Deze knop roept de administratorController aan die op zijn beurt het administratorModel instantieert en aanroept. De administratorController vraagt aan de Creator om de administrator klasse te instantiëren zodat de administrator klasse deze kan aanroepen.

Op het administrator scherm (administratorView) kan de administrator een gamecode genereren, het maximaal aan rondes voor de simulator invoeren, de beschikbare rollen beheren en de spelers zien die de deelnemen aan de simulator. De gamecode wordt getoond op de administratorView en de administrator geeft deze door aan de gebruikers die de simulator willen spelen. Wanneer de administrator de andere gegevens heeft ingevoerd en alle spelers worden weergegeven in de lijst op het administrator scherm kan de administrator de simulator starten.

Op dit moment wordt de RoleSelectionController aangeroepen. Dit object zal op zijn beurt weer de RoleSelectionModel instantiëren en aanroepen en de Creator klasse vragen om de RoleSelectionView te instantiëren. De speler kan op het rol selectie scherm een rol kiezen. In de lijst met rollen worden alleen de rollen getoond die door de administrator zijn goedgekeurd. De spelers kunnen alleen rollen uitkiezen die nog niet door een andere speler zijn uitgekozen. Wanneer de speler een rol kiest en deze wil bevestigen door op de knop te drukken wordt er een tekstveld met rode tekst weergegeven als de rol al gekozen is door een andere speler.

Als de speler een rol heeft gekozen die nog niet in gebruik is wordt de simulator getoond.

De simulator wordt gestart als de RoleSelectionController de SimulatorController heeft aangeroepen. De simulatorController instantieert en roept de SimulatorModel aan en vraagt de Creator om de simulatorView te instantiëren zodat de controller de view kan aanroepen.

Elke spel ronde van de simulator kan de speler acties uitvoeren door bijvoorbeeld een order op te vragen, een order te plaatsen of een order te versturen. De speler kan een beslissing maken over deze acties door op het scherm de weergegeven data af te lezen.

De simulator kan “buy orders” en “sell orders” afhandelen door het gebruik van het *chain of responsibility patroon*. Dit patroon kan door de Handler klasse verzoeken ontvangen van andere klassen en als dit mogelijk is handeld deze verzoeken af.

In de simulator is elke buy order of sell order een verzoek dat naar de handler verstuurd. De handler klasse zorgt ervoor dat de orders door de juiste klasse worden afgehandeld door een geïmplementeerde methode die is overgenomen uit de handler klasse. Elke speler heeft in de SimulatorModel een methode geïmplementeerd gekregen die de verzoeken van andere spelers kan opvangen en indien mogelijk afhandelen. Wanneer een order niet afgehandeld kan worden zal deze door de handler zelf afgehandeld worden. Op deze manier is er een gecentraliseerde en gestructureerde manier om orders van de verschillende spelers af te handelen.

Welke data getoond wordt aan de speler wordt bepaald door de geselecteerde rol van de speler. Na de spel ronde wordt alle data doormiddel van integers verstuurd naar de simulatorController klasse waar deze integers worden getest op de juiste invoer en range

van de integers. In geval van onjuiste invoer wordt de speler aangespoord om hun ingevoerde cijfers te veranderen.

Als de cijfers gevalideerd zijn worden deze naar de simulatorModel klasse verstuurd waar deze eveneens naar de database verstuurd zullen worden. Als de data succesvol is opgeslagen in de database wordt de volgende spel ronde gestart. Gebaseerd op de ingevoerde gegevens van de spelers zal de weergegevens data veranderen en de spelers gebaseerd op de nieuwe data beslissingen zullen moeten maken.

De speler kan op het simulator scherm de ordergeschiedenis bekijken door op de knop “orders” te drukken. De knop roept de OrderHistoryController aan die op zijn beurt weer de model en doormiddel van de Creator de OrderHistoryView instantieert en kan aanroepen.

De speler kan de gegevens die getoond worden op het ordergeschiedenis scherm niet veranderen en dus zal de orderHistoryView alleen data toegestuurd krijgen uit het model en niks versturen naar het model.

De gebruikersnaam van de speler wordt boven aan elke pagina weergegeven door de HeaderView. De header wordt bij elke view bovenaan weergegeven en toont de naam van de applicatie en de naam van de speler.

Wanneer de simulator de maximale ronde heeft bereikt zoals bepaald door de administrator zal de simulator beëindigen en wordt het scoreboard getoond. Aan het einde van de simulator wordt alle data opgeslagen in de database door het simulatorModel en wordt de ScoreboardController aangeroepen.

De scoreboardController instantieert het model en roept deze aan. Nadat de Creator de scoreboardView heeft geïnstantieerd en deze is aangeroepen wordt het scoreboard daadwerkelijk getoond. Het scoreboard scherm toont alleen data. Deze data kunnen niet worden aangepast op het scherm. Het scoreboard toont elke speler de belangrijkste data. Zo kunnen de interacties en beslissingen van de speler worden beoordeeld. De speler kan de applicatie nu sluiten indien gewenst.

5 Beveiliging

5.1 Veiligheid risico's

Voor het optimaal ontwikkelen van de applicatie doormiddel van Laravel en de programmeer talen PHP, HTML, CSS en SQL kunnen risico's in kaart worden gebracht door te kijken naar de data die wordt uitgewisseld tussen view, Controller en model. Het is de bedoeling dat de views het html en CSS-gedeelte omvat en dus minimaal logica bevat. De logica met betrekking tot de database wordt onderhouden in de models en logica die de afhandeling van exceptions, validatie en data stromingen van database naar front-end worden onderhouden in de controllers.

Valideren van gebruiker input:

De input van gebruikers is belangrijk. Als deze data niet wordt gecontroleerd op speciale tekens kunnen gebruikers SQL-injecties uitvoeren door bijvoorbeeld php code in een tekst vak te typen. Als schadelijke input zoals een injectie de database kan bereiken zou bijvoorbeeld een gebruiker de complete database buiten werking kunnen stellen.

Valideren van DB-data:

Het is een goede practice om data die al eerder in de controller gevalideerd was opnieuw te valideren en controleren op het toepassen van de juiste Mysql invoer types zoals string en integer en de lengte van bijvoorbeeld een string. Dit zorgt voor een tweede barricade tegen schadelijke code.

Scope en classes:

Het toepassen van private modifiers voor variabelen en properties is ook een goed idee om te voorkomen dat deze waardes kunnen worden aangepast buiten de scope van een classes of object. Dit geldt ook voor functies en methoden. Het gebruiken van setters en getters worden hierdoor noodzakelijk.

5.2 Oplossingen

Voor het optimaal beveiligen van de applicatie is het gebruiken van validatie in de controller classes en model classes aangeraden. Het gebruiken van afgeslote scopes binnen classes, functies en methode is ook aangeraden.

6 Prestatieoverwegingen

In dit hoofdstuk zal een lijst van beslissingen worden getoond waar samen met de opdrachtgever is gediscussieerd over mogelijke functionaliteit.

Prestatieoverwegingen:

- In overleg met de opdrachtgever is gekozen om de scope van het spel klein te houden en uitzonderlijk het logistieke proces van bierproductie bezig te zijn.
- In overleg met de opdrachtgever is gekozen om als eerst het simulator spel te ontwerpen met mogelijke uitbreidingen in de functionaliteit als de planning dit toelaat.
- In overleg met de opdrachtgever is gekozen om het ontwerp van *The beer game* te ontwerpen met twee parallelle usecases. Simulator spel(A) en simulator spel(B) zullen volledig worden uitgewerkt. De opdrachtgever kan dan kiezen welk ontwerp beter past bij zijn visie.
- In overleg met de opdrachtgever is gekozen om de tweede usecase simulator spel(B) verder uit te werken in schermontwerpen en diagrammen
- In overleg met de opdrachtgever is gekozen om af te zien van een inlogscherf. In plaats daarvan zal er een registratie pagina worden ontworpen.
- In overleg met de opdrachtgever is gekozen om een pagina te ontwerpen om een score aan de speler te tonen.
- In overleg met de opdrachtgever is gekozen om een pagina te ontwerpen om administrator functionaliteiten weer te geven.
- In overleg met de opdrachtgever is gekozen om een pagina te ontwerpen om de bestel geschiedenis van de speler weer te geven.
- Het ontwerp patroon *Singleton* is ontworpen voor de applicatie om de werking van het singleton patroon aan te tonen.
- Het ontwerp patroon *Factory method* is ontworpen voor de applicatie om de werking van de Factory method aan te tonen.
- Het ontwerp patroon *Chain of responsibility* is ontworpen voor de applicatie om de werking van de chain of responsibility aan te tonen.

7 Teststrategieën

Test scenario ID:	Test scenario	Verwacht resultaat	Acceptatiecriteria	Implementatiedatum
Test_01	PHP-unit test uitvoeren	Test zonder excepties uitgevoerd		
Test_02	Feature test uitvoeren	Test zonder excepties uitgevoerd		

Doormiddel van deze grafiek gaan wij onze applicatie testen. Elke functionaliteit die de applicatie moet bemachtigen noteren wij met een test scenario ID die er als volgt uit ziet: *test_01*. In de tweede kolom verwijzen we naar dezelfde test, maar dan met uitleg over de functionaliteit. Bij het verwachte resultaat noteren wij wat we denken dat er gaat gebeuren tijdens het testen. De acceptatiecriteria is het doel wat de functionaliteit moet behalen. De bovenstaande tabel wordt tijdens de realisatiefase van het project ingevoerd en bijgehouden.

De lijst met uitgevoerde testen zal bestaan uit feature testen en unit testen welke beiden door de editor, waar de applicatie in wordt ontwikkeld, wordt uitgevoerd. Door het gebruik van Laravel in de realisatiefase kan dit gemakkelijk geïmplementeerd worden.

In laravel kunnen unit testen en feature testen worden voorgeprogrammeerd in code. Deze testen worden uitgevoerd door het gebruik van de CMD (command line).

In laravel wordt een aparte test omgeving gebouwd wanneer de unit of feature testen worden uitgevoerd. De test omgeving kan door zijn eigen. Env bestanden worden geconfigureerd. Laravel maakt voor het testen van de gekoppelde database automatisch een test database aan waarop testen uitgevoerd kunnen worden.

8 Installatie- en implementatiehandleiding

De installatiehandleiding dient om mensen te helpen met het implementeren van de door ons ontwikkelde applicatie. Om de installatiehandleiding mogelijk te maken, gaan we een word tabel toevoegen op volgorde van alle stappen die genomen moeten worden.

De tabel zal in de realisatiefase worden ingevuld met de benodigde stappen voor het installeren en starten van de applicatie.

Stap	Handeling
Stap 1	Installeer Laravel en Composer
Stap 2	
Stap 3	
Stap 4	
Stap 5	
Stap 6	
Stap 7	

9 Onderhoud en toekomstige ontwikkeling

9.1 Afronding project

Tijdens de bouw van de applicatie hebben we een error gekregen dat het de `registrationModel` niet kon vinden. We hebben alles geprobeerd om het op te lossen, maar we hebben het niet op kunnen lossen. We moeten aan het eind van dit project dit project doorgeven aan een ander team zodat zij er verder mee kunnen gaan en de applicatie kunnen verbeteren en het probleem kunnen oplossen. Het doel van dit project is een inschatting van de productiviteit te maken. De applicatie zal alle activiteit van de computermuis kunnen vastleggen. Op deze manier zal de reactietijd van de speler samen met de productiviteit en genomen beslissingen zorgen voor een compleet overzicht van de spelers' capaciteiten.

9.2 Toekomstige taken en/of acties

In principe moet nog veel gedaan worden, dus wat er nog gerealiseerd moet worden zijn de registratie, de game zelf en alle functionaliteiten van het spel. Aangezien we al alle User story's hebben gemaakt kan het toekomstige team die dit project overneemt de User story's overnemen zodat zij minder werk te doen hebben.

Het realiseren van de ontwerp patronen is van groot belang om de applicatie op een gestructureerde manier te laten functioneren. De userstories uit de usecases kunnen letterlijk overgenomen worden en samen met de verschillende diagrammen kunnen de klassen worden opgebouwd zoals ontworpen. Voor een toekomstige uitvoering van dit project is het wel van belang om in een van de eerste realisatie sprints de ontwerp patronen te realiseren zodat dit niet in een later stadium wordt uitgevoerd en dan al bestaande klassen moeten worden aangepast.

Het is voor de database ook belangrijk dat een meer gedetailleerde ERD wordt opgesteld door het toepassen van normaliseren om tot een zo meest efficiënte database model te komen.

Tijdens het uitvoeren van de rest van de sprints moet de applicatie meermaals getest worden op de gerealiseerde functionaliteiten. Om gebruikers van de applicatie te helpen moet er ook een installatie en implementatie handleiding geschreven worden zodat gebruikers de applicatie op een efficiënte manier kunnen gebruiken.

10 Bronnenlijst

- Mota, M. (2024, 20 november). Model View Controller and Object Oriented Programming in PHP. *Medium*. <https://marmota.medium.com/model-view-controller-and-object-oriented-programming-in-php-638cecfa661e>
- *What is Class Diagram?* (Z.d.). <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/>
- Kilinc, B. (2024, 18 november). Mastering Design Patterns: A Deep Dive with Real-World Examples. *Medium*. <https://medium.com/turkcell/mastering-design-patterns-a-deep-dive-with-real-world-examples-c1eedfddcbbe>
- Kotsollaris, M. (2018, 15 mei). Applied Design Patterns & Software Architecture - Menelaos Kotsollaris - Medium. *Medium*. <https://mkotsollaris.medium.com/applied-design-patterns-software-architecture-42a8e6835cf0>
- Bloch, J., Garrod, C., & School of Computer Science. (2017). Patterns in 80 Minutes: a Whirlwind Java-centric Tour of the Gang-of-Four Design Patterns. In *15-214* (p. 1). <https://www.cs.cmu.edu/~charlie/courses/15-214/2016-spring/slides/24%20-%20All%20the%20GoF%20Patterns.pdf>
- Wikipedia contributors. (2025, 26 januari). *Design patterns*. Wikipedia. https://en.wikipedia.org/wiki/Design_Patterns
- Oodesign. (Z.d.). *Design patterns*. OODesign.com: Object Oriented Design. <https://www.oodesign.com/>
- Maheshmaddi. (2023, 11 april). 5. Concurrency patterns - Maheshmaddi - medium. *Medium*. <https://medium.com/@maheshmaddi92/5-concurrency-patterns-21adb11ee7c0>
- GeeksforGeeks. (2025, 2 januari). *Software Design Patterns tutorial*. GeeksforGeeks. <https://www.geeksforgeeks.org/software-design-patterns/>